SOLUTION TECHNIQUES FOR SINGLE-JOB LOT STREAMING PROBLEMS IN FLOW SHOPS

By

MEHMET AKANSEL

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

1998

I dedicate this work to my lovely wife, Neriman, and my adorable daughter, Sedef Ece.

ACKNOWLEDGMENTS

I would like to express my gratitude to my advisor, Dr. Süleyman Tüfekçi, who provided me with continuous support during my studies at the University of Florida. I also would like to thank him for setting a wonderful example for the academic life ahead of me.

I would also like to express my gratefulness to my beloved wife, Neriman, for standing by me during this journey and my sweet little daughter, Sedef Ece, for her big and bright smiles that boosted my spirit.

TABLE OF CONTENTS

ACKNOWLEDGMENTS iii

LIST OF FIGURES vi

LIST OF TABLES vii

ABSTRACT viii

1 INTRODUCTION 1

1.1 Characteristics of the Lot Streaming Problem 4
1.2 Classification of Lot Streaming Problems 7

2 LITERATURE REVIEW 11

2.1 Single Job Models 13
2.2 Multiple Job Models 20

3 SOLUTIONS OF PRIMARY LOT STREAMING PROBLEMS 24

3.1 LP Formulations of the $F, m \mid S, II \mid n, C, R \mid C_{max}$ Problem 24
3.2 Solution of the $F, 2 \mid S, II \mid n, C, R \mid C_{max}$ Problem 28
3.3 Solution of the $F, m \mid S, II \mid 2, C, R \mid C_{max}$ Problem 31
3.3.1 Baker and Pyke's Algorithm 31
3.3.2 Williams and Tüfekçi's $O(m^2)$ Algorithm 36
3.3.3 Williams and Tüfekçi's Linear-Time Algorithm 41
3.4 Solution of the $F, 3 \mid S, II \mid n, C, R \mid C_{max}$ Problem 46
3.5 Solution of the $F, m \mid S, II \mid 3, C, R \mid C_{max}$ Problem 51
3.6 Solution of the $F, m \mid S, II \mid n, C, R \mid C_{max}$ Problem 62
3.6.1 Williams, Tüfekçi and Akansel's Heuristic 62
3.6.2 An $O(n^3 * (m + n))$ Algorithm for the m x n Problem 66
3.6.3 An $O(n^2 * (m + n))$ Algorithm for the m x n Problem 98

4 EXTENSIONS TO THE BASIC LOT STREAMING MODEL 104

4.1 Lot Streaming Problems with Setups Explicitly Considered 104
4.1.1 Lot Streaming Problems with Detached Setups 105
4.1.1.1 Two-sublot problem with detached setups 109
4.1.1.2 Three-sublot problem with detached setups 113
4.1.1.3 General size problem with detached setups 123
4.1.2 Lot Streaming Problems with Attached Setups 141
4.1.2.1 Two-sublot problem with attached setups 142
4.1.2.2 Three-sublot problem with attached setups 145
4.1.2.3 General size problem with attached setups 155
4.2 Lot Streaming Problems with Transfer Times Explicitly Considered 161

5 CONCLUSION 165

REFERENCES 167

BIOGRAPHICAL SKETCH 171

LIST OF FIGURES

Figure 1.1 Lot streaming with equal sublots and intermittent idling 4

Figure 3.1 Network representation of the m machine n sublot problem 33

Figure 3.2 Network representation of the m machine 2 sublot problem 34

Figure 3.3 Representation of the m machine 2 sublot problem in $\mathcal{R}^2$ 44

Figure 3.4 Network representation of the m machine 3 sublot problem 53

Figure 3.5 Network representation of the m x n lot streaming problem....67

Figure 4.1 Network representation of the lot streaming problem with detached setups 108

Figure 4.2 Network representation of m x 2 problem with detached setups 110

Figure 4.3 Network representation of m x 3 problem with detached setups 114

Figure 4.4 Network representation of the two sublot problem with attached setups ...144

Figure 4.5 Network representation of the three sublot problem with attached setups .147

Figure 4.6 Network representation of m x n problem with constant transfer times. 163

LIST OF TABLES

Table 4.1 Performance evaluation of the Bisection - DSA 122

Table 4.2 Performance evaluation of the 3-sublot Bisection - ASA 154

Table 4.3 Performance evaluation of the *n*-sublot Bisection - ASA 160

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

SOLUTION TECHNIQUES FOR
SINGLE-JOB LOT STREAMING PROBLEMS IN FLOW SHOPS

By

Mehmet Akansel

May, 1998

Chairman: Dr. Süleyman Tüfekçi
Major Department: Industrial and Systems Engineering

Lot streaming can be defined as the policy of allowing some portions of the production lot to move to downstream machines before the whole production lot completes its processing on the upstream machines. This action results in overlapped processing periods for the same production lot on a number of sequential work stations which eventually reduces the time the lot spends in the shop. Hence, the application of this idea in various manufacturing environments can provide improvements in many time-based performance criteria such as makespan and total flow time. Meanwhile, since the production lot starts "streaming" through the system, there will be less accumulation in the buffers, that is, work-in-progress (WIP) inventory will decrease. Lot streaming is a technique that can be successfully implemented in conjunction with some modern manufacturing technologies, such as Just-in-Time (JIT), Flexible Manufacturing Systems (FMS) and Group Technology (GT). However, lot streaming can also be applied in

conventional manufacturing environments where production capacity does not justify any kind of automation.

Although the idea of splitting the production lot in order to achieve overlapped processing times with the ultimate goal of reducing the flow time and WIP inventory has been long known to the manufacturers, the efforts to obtain the maximum advantage from this policy is quite new. In the relatively short history of lot streaming area, many lot streaming problems have been introduced. In this dissertation, several lot streaming problems are introduced and solution techniques developed for these problems are presented. The contribution of the author includes two efficient algorithms with complexities of $O(n^3*(m+n))$ and $O(n^2*(m+n))$, respectively, for the basic lot streaming problem where a single-job production lot can be split into any number of consistent sublots in a flow shop with any number of machines. The objective of makespan minimization was used in every lot streaming problem studied in this work while intermittent idling is allowed on the machines. The author has also developed near-optimal solution techniques for the lot streaming problems with setups included. The robust performance of the algorithms is supported by extensive experiments in which the deviation from the optimal solution rarely exceeds 1 percent.

CHAPTER 1
INTRODUCTION

In today's world, the key to the success of a manufacturing company lies in the practices that provide the customers with the quality products in a cost-effective and timely manner. Therefore, the practitioners who want to have a better edge in the merciless global competition have been using several different approaches to achieve this ultimate goal. Material Requirements Planning (MRP) is one of the earliest computer-based systematic approaches initiated to help the manufacturers satisfy the customers' demands in a more timely manner. MRP is designed to calculate order release dates for components, so that their production can be started early enough to guarantee that the required end products will be ready to ship on given due dates. For this purpose, the system assumes that there is a known fixed lead time between an order release and its completion. With this logic, it schedules all activities backwards from the due dates. The result is often an infeasible schedule that causes unbalanced utilization of the resources. Since the entire lot is treated as a single unit while processing on a sequence of work stations, large work in progress (WIP) inventories and long cycle times are major drawbacks in the MRP systems. Hence, in order to reduce WIP and cycle time, some new manufacturing schemes were introduced.

Just In Time (JIT), Group Technology (GT) and Optimized Production Technology (OPT) have emerged as more realistic answers to long unresolved issues in manufacturing. These are not decision support systems as MRP intended to be, but rather

production flow schemes designed to improve performance of the manufacturing plants. JIT aims to reduce WIP while keeping the deliveries timely. The implicit principle of JIT is to manufacture the products in unit-size sublots and let sequential manufacturing activities overlap on downstream work stations, so that the products will be ready in the shortest possible amount of time. However, in certain manufacturing environments having unit-size sublots may not always be feasible due to the high transportation expenses between work stations. Therefore, sometimes sublot sizes larger than unity may be a necessity even in a JIT environment. The components of the system must be kept under strict coordination for a successful JIT implementation. Otherwise, if a component fails to deliver the materials required by downstream elements, the throughput of the system will deteriorate significantly since buffers are minimized to reduce the WIP inventories. Another requirement by JIT is that setup times must be reduced significantly in order to keep the system active with smaller lots and reduced buffer sizes.

Group Technology (GT), which is another modern manufacturing strategy, is a plant layout and organization scheme. The work stations are located so that typical products can flow through one of the several prearranged routes. The main advantage of GT, comparative to the regular flow shop arrangement, is to reduce the cost of transferring jobs or lots from one station to the next. This makes it possible to use small transfer lots, and thus, operate in a "JIT-like" manner.

Optimized Production Technology (OPT) has emerged as a commercial software-based technique that applies the main JIT principles to a batch production environment. OPT specifies the required buffer sizes for the bottleneck stations and allows the splitting of batches on these stations in order to improve the utilization of the downstream stations

and reduce WIP inventories and cycle times. Based on the applications of OPT, its developer, E.M. Goldratt, generated several logistics techniques and principles which are the tools of the management philosophy called Theory of Constraints (TOC) [Gol92, Cox97]. The TOC philosophy is based on the principle of focusing attention on those few elements, which are the constraints of the system, that control the performance of the entire system. One of the main tools of the TOC is called the Drum-Buffer-Rope (DBR) method. DBR is a scheduling method that binds the flow of all materials to the capacity of the constraint. The constraint is called the drum, and it determines the rate of the throughput for the line. The buffer is the amount of time between the release of material to the first work station and its scheduled processing time at the constraint. The rope is the policy of releasing the material at the first station based on the schedule of the constraint, that is, the beat of the drum. Meanwhile, DBR arbitrarily determines sizes of the transfer batches, which are smaller than the process batch, between nonconstraint work stations so that processing times can be overlapped and the time to reach the constraint, i.e., the buffer, can be reduced.

In all the methods mentioned above, one can employ the concept of lot streaming. Lot streaming can be simply defined as the process of splitting a job lot or batch into sublots or transfer batches so that its operations on subsequent work stations can be overlapped. This simultaneous processing of sublots results in reductions in WIP as well as in lead times and makespan as shown in Figure 1.1. A 20-unit lot is split into two equal sublots in a three machine flow shop where unit processing times are 2, 1, and 2 time units, respectively. Although the resulting schedule is not optimal, the makespan

decreases to 70 time units as opposed to 100 time units (= 20 * (2 + 1 + 2)) in the case without lot streaming.

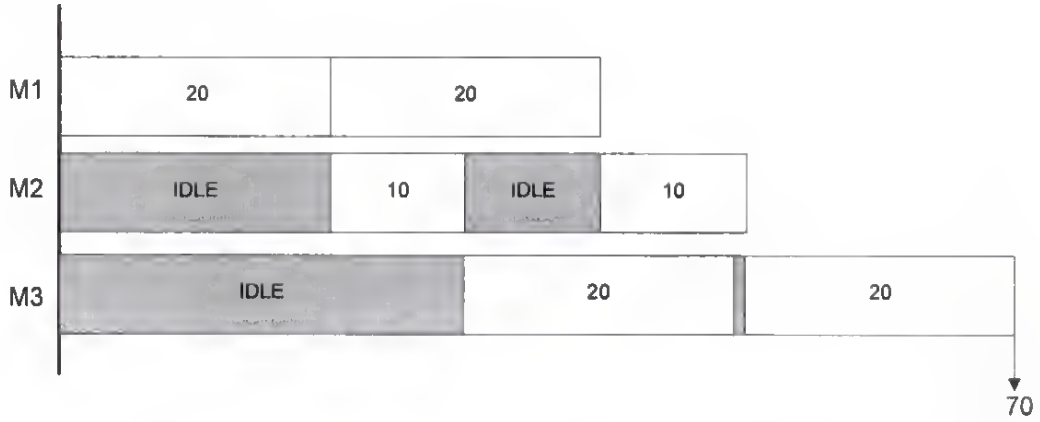

Figure 1.1 Lot streaming with equal sublots and intermittent idling.

The benefits of lot streaming have long been known by the practitioners and applied in an intuitive manner. Although the concept of lot streaming was first introduced formally by Reiter in 1966 [Rei66], the research focused on finding optimal solutions to lot streaming problems only after late '80s. Nonetheless, many aspects of lot streaming have stayed open for further research. In order to establish a basic ground for further discussion on the topic, a problem classification is proposed in the next section.

1.1 Characteristics of the Lot Streaming Problem

Similar to the conventional scheduling theory, there are several factors to be used in the categorization of lot streaming problems. Depending on the manufacturing environment, each characteristic assumes one of various settings available. In order to be able to define a classification of lot streaming problems, it is beneficial to review the characteristics that are frequently observed in the industry.

Process type (*flow shop / job shop / open shop*): This characteristic is mainly concerned with the layout of the manufacturing shop. Since this directly affects the route that the production items follow in the shop, type of the process is very important when formulating the lot streaming problem. In the previous research, the most studied process type has been the *flow shop* arrangement, where all items follow the same sequence in the strict order and each machine represents a different manufacturing stage. Therefore, the terminology and problem classification in this part of the lot streaming are very close to the flow shop scheduling terminology. Meanwhile, lot streaming can also be applied in *job shops*, where some distinct manufacturing stages can be performed on the same machine, that is, a machine can be visited more than once, and *open shops*, where manufacturing stages do not have any precedence relationships and therefore can be performed in any order [Ste93, Gla94].

Number of machines (*m*): Number of machines in the shop that the production items need to visit is very important in terms of the solvability of the problems. Similar to the regular flow shop scheduling, where problems having more than two machines are *NP-hard* [Pin95], some lot streaming problems require special attention when their machine number grows considerably.

Number of jobs (*Single/Multiple*): A large portion of the lot streaming literature has dealt with single job systems which do not have much meaning in the scheduling theory. However, multi-product lot streaming problems have also been studied by some researchers to a limited extent.

Number of sublots (n): Number of sublots is sometimes predetermined. However, in some models there may be only an upper bound on the number of sublots due to some capacity limitations of the manufacturing system.

Discrete or continuous sublot size: In some certain applications, sublot sizes may be restricted to assume only integer or *discrete* quantities. Typically, this discrete version of the problem can be formulated as an integer linear program. Since it is expected that this may become a difficult problem to solve, sometimes *continuous* version, where integrality constraints are relaxed, can be used to obtain some insights and occasionally reasonable solutions.

Variable or consistent sublots: This item is related to the flexibility of the manufacturing environment. In some environments, sublots may be allowed to change their sizes at some point in the manufacturing process. This may help increase the machine utilization by reducing idle periods between sublots. When the sublots between machines i and (i+1) are allowed to differ from the sublots between machines (i-1) and i, the model will have *variable* sublots. However, in some manufacturing environments, this policy is not cost effective and the sublots are kept the same all across the manufacturing plant. In this case, the sublots are *consistent*.

Intermittent idling or no idling: In some environments, it may be required that a work station continue to be active once started and it must process the entire lot continuously, without idling. This may be a result of the high start-up and operating costs for the machine. Therefore, any idle time on the machine is detrimental to the profitability of the process. When there is such a restriction, the start-up time of a machine may be delayed in order to eliminate the idle periods between sublots, and therefore, overall

completion time will probably rise. If there is no such restriction, idling may occur between sublots and/or jobs, that is, the schedule will have *intermittent idling*.

Setups: In some lot streaming models, setups may be assumed to be included in the processing times or may not be a consideration at all. However, when it is important to consider setups independently, one of the following two cases will be observed: *attached setup* or *detached setup*. If setup activities cannot be performed until some work pieces arrive at the machine that will be prepared for the processing, the setups are said to be *attached*. Attached setup activities may involve some measurements on the work piece and adjustments on the tool and fixture. If setups can be completed as soon as the machines become available, they are called as *detached* setups. While there may be pure cases where all machines having the same type of setup in the system, it may be possible to observe a mixture of attached and detached setups on different machines in the same shop.

Performance criterion: There are several performance criteria that have been used in scheduling theory, such as makespan, which is the completion time of the last job or sublot on the last machine, total completion time, weighted completion time, maximum tardiness, and so on. Makespan minimization has been the most frequently used objective function in lot streaming problems. However, other performance criteria used in scheduling are also applicable in lot streaming problems.

1.2 Classification of Lot Streaming Problems

After the introduction of the characteristics in the previous section, the following classification is proposed for lot streaming problems. A four-field problem descriptor will

be helpful to express the significant characteristics of the model. In this four-field notation, a problem type is represented by $\alpha \mid \beta \mid \gamma \mid \lambda$, where α represents the manufacturing environment, β defines characteristics of the job, γ denotes characteristics of the sublots, and λ is the objective function. Let ° denote the empty symbol.

The first field takes the form $\alpha = \alpha_1\alpha_2$, where α_1 and α_2 are represented as follows.

- $\alpha_1 \in \{F, J, O\}$:
  - $\alpha_1 = F$: a flow shop;
  - $\alpha_1 = J$: a job shop;
  - $\alpha_1 = O$: an open shop.
- $\alpha_2 \in \{k, m\}$:
  - $\alpha_2 = k$: specified machine number, where k is a constant positive integer;
  - $\alpha_2 = m$: general machine number.

The second field takes the form $\beta = \beta_1\beta_2\beta_3$ where β_1, β_2 and β_3 are represented as follows.

- $\beta_1 \in \{S, P\}$:
  - $\beta_1 = S$: single job in the system;
  - $\beta_1 = P$: multiple jobs in the system.
- $\beta_2 \in \{NI, II\}$:
  - $\beta_2 = NI$: no idling is allowed;
  - $\beta_2 = II$: intermittent idling is allowed.
- $\beta_3 \in \{°, AS, DS\}$:

- $\beta_3 = \circ$: setups are not explicitly considered;
- $\beta_3 = AS$: setups are attached to the first sublot;
- $\beta_3 = DS$: setups are detached.

The third field takes the form $\gamma = \gamma_1\gamma_2\gamma_3$ where γ_1, γ_2 and γ_3 are represented as follows.

- $\gamma_1 \in \{l, n\}$:
  - $\gamma_1 = l$: specified sublot number, where l is a constant positive integer;
  - $\gamma_1 = n$: general sublot number.
- $\gamma_2 \in \{C, E, V\}$:
  - $\gamma_2 = C$: sublots are consistent all across the system;
  - $\gamma_2 = E$: sublots are equal to each other all across the system;
  - $\gamma_2 = V$: sublots are variable.
- $\gamma_3 \in \{I, R\}$:
  - $\gamma_3 = I$: sublot sizes are restricted to be positive integers;
  - $\gamma_3 = R$: sublot sizes may assume real values.

Lastly, the objective function field may assume one of the following.

- $\lambda \in \{C_{max}, L_{max}, \Sigma C_i, \Sigma w_i C_i\}$:
  - $\lambda = C_{max}$: minimize the makespan (maximum completion time);
  - $\lambda = L_{max}$: minimize maximum lateness;
  - $\lambda = \Sigma C_i$: minimize total completion time (of jobs and/or sublots);
  - $\lambda = \Sigma w_i C_i$: minimize total weighted completion time (of jobs and/or sublots).

Using the scheme above, a single-job three-machine flow shop lot streaming problem using makespan criterion which assumes four consistent, integer-valued sublots, allows intermittent idling and considers no setups on the machines can be represented as $F, 3 \mid S, II \mid 4, C, I \mid C_{max}$.

The themes of the next chapters are briefly the following. Chapter 2 provides a comprehensive literature survey in the lot streaming area. Chapter 3 discusses the solution techniques offered for single-job lot streaming problems in flow shops. This part also contains the accomplishments on m machine 3 sublot and m machine n sublot problems. Chapter 4 studies the lot streaming models extended to include setup activities on the work stations and transportation delays between the work stations. This chapter also provides robust algorithms for lot streaming problems with both attached and detached setups. Finally, Chapter 5 includes the summary of the results achieved in this dissertation.

CHAPTER 2
LITERATURE REVIEW

Although the benefits of lot streaming has been realized by practice for a long time, there was limited work in the literature on obtaining optimal solutions to any lot streaming model until late '80s. The term "lot streaming" was introduced by Reiter in 1966 [Rei66], but the idea has been considered many times under different names. Several authors have investigated the effects of lot streaming and concluded that it offers significant improvement in job timeliness at the expense of a modest increase in transportation cost [Han71, Pot92].

One of the earliest studies on lot streaming was done by Szendrovits [Sze75]. In his work, he analyzed the lot streaming problem in a flow shop for a single job with equal sublot sizes ($F, m \mid S, NI \mid n, E, R \mid C_{max}$) and investigated relationships among the lot size, the manufacturing cycle time and average work-in-progress inventories. He showed that manufacturing operations can be overlapped through the use of subbatches, thereby manufacturing cycle-time is shortened, and additionally, total production and inventory carrying costs are reduced compared to the traditional EOQ (economic order quantity) model. Goyal [Goy76] extended Szendrovits' model to include fixed cost of moving a subbatch through all the machines and proposed an iterative approach to obtain optimal solution to the new model using partial derivatives of the functions developed to define total production and inventory carrying cost. However, following Goyal's work on his model, Szendrovits [Sze76] first stated that number of subbatches must be determined in

advance when transportation cost for each subbatch is hard to obtain. He also developed a simpler approach to obtain the optimal subbatch number and lot size under Goyal's assumptions.

Truscott [Tru85] developed an algorithm to generate the best schedule with equal size subbatches as suggested by Szendrovits. In this work, he classified the production activities into two groups such as 'operations,' whose completion time is proportional to the subbatch size, and 'complements,' whose completion time is independent of the subbatch size. He did not allow any idle periods between subbatches. He also added a transportation time for each subbatch moving between work stations. In a later study, Truscott [Tru86] improved his model to include constraints on the transportation activities between work stations. In this extension, he introduced two types of constraints. The first set specifies a maximum number of units per load for each transportation activity; the second one imposes a limit on the availability of vehicles for moving loads between operations. Then, he developed some algorithms that schedule the subbatches without idle periods in between, while obeying the new capacity constraints.

Graves and Kostreva [Gra86] suggested that overlapping operations in MRP provide an enhancement that allows wider applicability, shortened lead times, and lower total costs. Considering two work stations with equal production rates, they developed closed-form formulas for lead time and total production cost to be used in the MRP software utilizing overlapping policy. Under these assumptions, they calculated the production lot size and the number of subbatches between the two particular work stations.

By the end of the '80s, an increased interest was observed on the topic. This is because it is consistent with the JIT philosophy of making small or unit size sublots and it also agrees with the basic idea of OPT. Thereafter, two research paths have been observed in the lot streaming literature. The first one assumes that the numbers of sublots are predetermined, and then examines the best way to allocate work to this given number of sublots. The focus is on the optimization of a single job lot. The second path generally assumes there is no limit on the number of sublots, so that unit size sublots are optimal. In this case, allocation is not generally an issue, and the analysis focuses on the sequencing of multiple jobs. The literature on single and multiple job lot streaming problems will be reviewed in two separate sections below.

2.1 Single Job Models

Baker [Bak87] and Trietsch [Tri87] independently developed models to study single job lot streaming in a two-machine flow shop ($F, 2 \mid S, II \mid n, C, R \mid C_{max}$). Baker formulated an LP model regarding the capacity constraints on the machines and technological constraints on the job and used makespan as the performance criterion. He also developed an alternative formulation where the objective is to minimize idle periods occuring in the schedule. As shown by Baker, optimal sublot sizes in two machine problems are obtained through the following geometric formulas,

$$x_j = p^{j-1} / (1 + p + p^2 + ... + p^{n-1}), \quad \text{for } 1 \le j \le n,$$

while the optimal makespan is

$$M^* = (1 + p + ... + p^n) / (1 + p + ... + p^{n-1}),$$

where $p = p_2 / p_1$, and p_i, $i = 1, 2$, represents the unit processing time on machine i and x_j is the fraction of the lot allocated to sublot j. He also commented on two characteristics of this solution. First, the optimal allocation of work among the sublots is uneven. In particular, if the second machine is faster than the first, then the sublot sizes should be decreasing; if the second machine is slower, then the sublot sizes should be increasing. Second, the schedule is shortened as the number of sublots increases. Thus, he suggested that the ideal number of splits is infinite. In this study, Baker also provided some insights to three-machine, two-sublot cases using his alternative linear programming formulation. In this case, the solution will be affected by the magnitude of processing time on the second machine relative to the other two. Namely,

- if $p_2^2 - p_1 p_3 > 0$ and $p_1 > p_3$, $x_1 = p_1/(p_1 + p_2)$, and $x_2 = p_2/(p_1 + p_2)$, or
- if $p_2^2 - p_1 p_3 > 0$ and $p_3 > p_1$, $x_1 = p_2/(p_2 + p_3)$, and $x_2 = p_3/(p_2 + p_3)$, or
- if $p_2^2 - p_1 p_3 < 0$, $x_1 = (p_1 + p_2)/(p_1 + 2p_2 + p_3)$, and

  $x_2 = (p_2 + p_3)/(p_1 + 2p_2 + p_3)$.

Meanwhile, Trietsch used a heuristic approach to study the same 2-machine environment and obtained the same geometric formulas. He also provided a simple technique to obtain integer sublot sizes. He showed that the maximum marginal gain is obtained with the first extra transfer and the number of sublots will be practically bound from above by $O(\log(m))$ as opposed to Baker's suggestion of unlimited number of sublots.

Trietsch [Tri89] later extended his studies to cover flow shops having more than two machines ($F, m \mid S, II \mid n, V, I \mid C_{max}$) and included budget constraints on transfer

activities between consecutive machines. In this study, he developed a method to recognize bottleneck machines and let sublot sizes vary between pairs of subsequent bottleneck machines, therefore, sublot sizes may not be consistent.

Potts and Baker [Pot89] showed that consistent sublot sizes provides the shortest makespan when there are two or three machines in the system. They showed by an example that variable sublot sizes result in smaller makespan when machine number exceeds three. Subsequently, they provided upper bounds on the effect of equal sublot assumption over the makespan compared to the consistent sublot case.

Kropp and Smunt [Kro90] provided lot streaming formulations for single job lot streaming problems in a flow shop. They also extended their models to include attached setups, transportation times between work stations, and WIP related performance measures ($F, m \mid S, II, AS \mid n, C, R \mid C_{max}$). They also presented some experiments in order to examine the effect of setup times over the sublot sizes. According to this study, when the ratio of the setup times to the processing times is high, the first sublot becomes very small in order to finish the setups as early as possible while the rest of the sublots share the remaining of the lot almost equally.

Baker and Pyke [Bak90] developed a network representation for the single job lot streaming problem ($F, m \mid S, II \mid 2, C, R \mid C_{max}$). They used an activity-on-node diagram whose nodes represent the processing of individual sublots on each machine. The makespan corresponds to the longest path in the network, however, since sublot sizes are decision variables, this network has variable-length activity times. Therefore, the problem of minimizing the makespan involves allocating work to sublots to minimize the length of the longest path in the network. They proposed a solution for the two sublot problem

based on the claim that the machine with the largest processing time is always on the longest path in the network. However, this is not always the case as shown by Williams and Tüfekçi [Wil92]. Therefore, their algorithm, which has $O(m)$ complexity, often results in suboptimal solutions.

Williams and Tüfekçi [Wil92] developed $O(m^2)$ algorithms for m machine 2 sublot problems (F, m | S, II | 2, C, R | C_{max}) after showing Baker and Pyke's algorithm does not always find the optimal solution. They used the network representation of the problem to generate efficient algorithms. In the network representation, the longest path is also called the *critical path* and the machine index i, $1 \leq i \leq m$, where the critical path connects two horizontally adjacent nodes such as (i, 1) and (i, 2), is named as the *critical index*, or *crossover index*. Before starting to search for the longest path all over the network, they determine the machine indices with a possibility of being on the critical path. These machine indices are called the *feasible indices*. Using a feasible index search algorithm, they eliminate the machines that may never be on a critical path. They showed that the slowest machine is always a feasible index but not necessarily a critical index.

Trietsch and Baker [Tri92] gave a classification of lot streaming problems and summarized their independent achievements in their joint work. They provided a dominance diagram associated with the problem classes. According to this comparison, the problem with variable sublots and intermittent idling assumption is the least restricted one, while equal sublots and no idling assumption results in the largest makespan. They also studied a two-machine problem with limited transporter capacity.

Baker and Jia [Bak93] studied numerical effects of various assumptions in three machine lot streaming problems (F, 3 | S, NI/II | n, $C/E/V$, R | C_{max}). Their findings

confirm that variable sublots result in shorter makespan compared to the consistent sublots. They observed that no-idling assumption causes 50 % increase on the makespan when the second machine is dominated, i.e., $p_2^2 < p_1 p_3$. They stated that some redundancy occurs depending on the dominance of the second machine. When the second machine is dominant, i.e., $p_2^2 \geq p_1 p_3$, the no-idling constraint becomes redundant since it is satisfied automatically; when the second machine is dominated, i.e., $p_2^2 < p_1 p_3$, the consistent sublots constraint becomes redundant.

Steiner and Truscott [Ste93] studied an open shop lot streaming problem where sublots are equal and machines are restricted to operate continuously ($O, m \mid S, NI \mid n, E, R \mid C_{max}$). They stated that when the operations can be performed in any order in the shop, the problem of minimizing the makespan becomes very closely related to a traveling-salesman problem. They showed that unit sublot sizes are optimal in both makespan and total flow time minimization cases.

Glass, Gupta and Potts [Gla94] developed efficient solution approaches to three machine flow shop, job shop and open shop lot streaming problems. In three machine flow shop ($F, 3 \mid S, II \mid n, C, R \mid C_{max}$), similar to the limited findings of Baker [Bak87], they showed that, when the second machine is dominated, the ratio of two consecutive sublot sizes will be equal to the ratio of sum of the processing times on the first two machines to sum of the processing times on the last two machines. However, when the second machine is dominant, the solution scheme will be slightly difficult. In the optimal solution, up to a sublot k, the ratios will be determined using the processing times on the first two machines, while thereafter only the last two machines will be used. Therefore,

finding the change-over sublot k becomes important, which can be done in O(log (n)), where n is the number of sublots.

Chen and Steiner [Che94] improved Glass et. al.'s three-machine flow shop model by adding attached setups where setups on each machine require the first sublot arrive from the previous machine in order to be completed (F, 3 | S, II, AS | n, C, R | C_{max}). They characterized the possible critical paths in the project network, and determined optimal sublot sizes in O(n) time complexity level, where n is the number of sublots, by checking all possible paths.

Chen and Steiner [Che95] used the fundamentals developed by Glass et. al. for three-machine flow shop model in order to extend the model to include detached setups where setups on each machine can be completed without waiting for the first sublot to arrive from the previous machine (F, 3 | S, II, DS | n, C, R | C_{max}). Similar to their attached setup model [Che94], they characterized the possible critical paths in the project network, and obtained optimal sublot sizes in O(log(n)) time for this case, since the problem provides the required monotonicity for bisection search, where n is the number of sublots.

Çetinkaya and Gupta [Cet94a] adopted total flow time and total weighted flow time as their performance criteria in single-job flow shop lot streaming problems (F, m | S, II | n, C, R | ΣC_i / $\Sigma w_i C_i$). They showed that m machine n sublot total flow time problems are quite simple, where all sublots except the last one are equal to the minimum amount specified which is usually 1. However, arbitrary machine and sublot number total weighted flow time problems can be solved efficiently only when the first machine is the slowest. They also developed efficient algorithms for m machine 2 sublot and 2 machine n sublot problems in total weighted flow time cases with consistent and variable sublots.

Benli [Ben94] proposed a model in which lot sizing and machine sequencing activities are integrated. He adopted a periodic review model with variable period lengths and suggested several objective functions available for the model which includes the inventory balance equations, capacity constraints, sublot number restrictions and transfer time specifications.

Şen, Topaloğlu, and Benli [Sen94] studied a single job two-machine flow shop lot streaming problem under several assumptions. They first summarized the consistent and equal sublot version of the makespan minimization problem ($F, 2 \mid S, II \mid n, C/E, R \mid C_{max}$). They also commented on how to formulate a tardiness minimization problem ($F, 2 \mid S, II \mid n, C, R \mid L_{max}$). Subsequently, they developed solution schemes under sublot completion and item completion assumptions ($F, 2 \mid S, II \mid n, C/E/V, R \mid \Sigma w_i C_i$). According to their study, when the first machine is slower than the second machine, equal sublots on each machine minimize total weighted flow time. However, when the second machine is the slowest, optimum solution is obtained by geometric sublots on the first machine and equal sublots on the second machine. Topaloğlu, Şen and Benli [Top94] also developed efficient algorithms for two consistent sublot total weighted flow time minimization problem ($F, m \mid S, II \mid 2, C, R \mid \Sigma w_i C_i$) under sublot and item completion assumptions.

Williams and Tüfekçi [Wil95] solved m machine 2 sublot makespan minimization problem ($F, m \mid S, II \mid 2, C, R \mid C_{max}$) optimally in linear time. They employed a search method that was developed by Megiddo [Meg83] originally for solving linear programs with 2 decision variables and m constraints. In this implementation, they obtained the optimal solution without referring to the feasible index search concept. They essentially generate all linear functions of makespan associated with each machine index, and then,

look for the minimizing point over the convex region which is created by the maximum of all functions.

Williams, Tüfekçi and Akansel [Wil97] developed an efficient algorithm for m machine 3 sublot problem (F, m | S, II | 3, C, R | C_{max}). This algorithm uses partial derivatives of the objective function in terms of the ratios of the consecutive sublot sizes. Since the objective function is convex in $\mathfrak{R}^3$, the sign of the partial derivatives leads to the global minimum. Williams et. al. also stated that the complexity of their algorithm is $O(m)$, when the feasible indices are known. However, if the feasible indices also need to be determined, the overall complexity becomes $O(m^2)$ due to the Feasible Index Search Algorithm. They also devised a heuristic to solve n sublot problem with an approximation error less than 1 % by using the optimal solution obtained for the 3 sublot problem. The complexity of the heuristic algorithm for the m machine n sublot problem is $O(mn^2)$.

2.2 Multiple Job Models

Multiple job models mainly deal with integrating lot sizing and scheduling. In one of the earliest works, Jacobs and Bragg [Jac88] used a simulation model to examine the lot streaming principle and its effect on flow times in a flow shop. They used the concept *repetitive lots* instead of lot streaming for the same idea. In their model, when a work station finishes processing on a sublot, the scheduling rule gives priority to another sublot of the same product. Their simulation experiments demonstrate that lot streaming can substantially reduce mean flow times compared to the ones obtained without lot streaming.

Potts and van Wassenhove [Pot92] reviewed the literature on batching and lot sizing and stated that these decisions can be integrated with scheduling decisions. They briefly summarized the existing single job lot streaming research and commented that multiple job lot streaming problems were open to further research.

Potts and Baker [Pot89] studied the multiple job flow shop lot streaming problem ($F,\ m \mid P,\ II \mid n,\ C,\ R \mid C_{max}$) and showed that this problem cannot be solved by first constructing an optimal sequence and then independently splitting each job into optimal sublots. This means that optimal solution will not be obtained unless preemption of a job or intermingling of sublots of different jobs is allowed.

Çetinkaya and Kayalıgil [Cet92] developed a multiple job lot streaming model for a two machine flow shop where preemption of a job is not permitted and setups are detached from the first sublot ($F, 2 \mid P,\ II,\ DS \mid n,\ E,\ I \mid C_{max}$). They assumed there were no limitations on the movement of unit sized sublots. Showing that unit sublots were optimal, they aimed to achieve a model where they can apply the renowned Johnson's rule [Joh54] for two machine flow shops. For this purpose, they developed *run-in time* and *run-out time* concepts which can be described as the latest delay time of a setup for a job j at the second machine without affecting the completion of the job and the difference between the completion times of job j at the first machine and the second machine, respectively. Afterwards, they applied the Johnson's rule solely based on the run-in and run-out times associated with each job.

Vickson and Alfredsson [Vic92] studied a similar model where the setups are not considered ($F, 2 \mid P,\ II \mid n,\ E,\ I \mid C_{max}$). They conjectured that when solving a two machine problem, scheduling must be made using the unit processing times instead of the lot

processing times with the Johnson's rule. They claimed that even when a batch move time is added in the model, the same solution with zero move-times would be optimal. In the three-machine case, they developed some solution schemes based on a comparison between the processing times on the second machine and the processing times on the first and third machines. In this case, the solution will be obtained through the following ways.

(a) If $p_{i1} \geq p_{j2} \; \forall \; i, j$ or $p_{i3} \geq p_{j2} \; \forall \; i, j$, the problem has the same solution as the two-machine problem with unit processing times (P_{i1}, P_{i2}) given as $P_{i1} = p_{i1} + p_{i2}$ and $P_{i2} = p_{i2} + p_{i3}$;

(b) If $p_{i2} \geq p_{j1} \; \forall \; i, j$, the problem can be solved by letting each of the N jobs be first, sequencing the remaining (N-1) jobs using the Johnson's rule with processing times (p_{i2}, p_{i3}), and then by choosing the best of the resulting N schedules; and

(c) If $p_{i3} \leq p_{j2} \; \forall \; i, j$, the problem can be solved by letting each of the N jobs be last, sequencing the remaining (N-1) jobs using the Johnson's rule with processing times (p_{i1}, p_{i2}), and then by choosing the best of the resulting N schedules.

Dauzère-Péres and Lasserre [Dau93] considered multiple job lot streaming problem in a job-shop environment where they aimed to simultaneously determine the sublot sizes and sequences of the sublots on the machines ($J, m \mid P, II \mid n, C, R \mid C_{max}$). Their procedure alternates between solving a *lot streaming procedure* with a fixed number of sublots and a fixed sequence of sublots on the machines and a *scheduling algorithm* with a fixed number of sublots and a fixed size for each sublot. They commented that lot streaming provides significant improvement over makespan even

with a few sublots. They suggested that when production lot becomes larger, number of sublots must also be increased.

Çetinkaya [Cet94b] studied a multiple job model in a two machine flow shop with separate processing, setup and transfer times specified for each job (F, 2 | P, II, DS | n, V, R | C_{max}). In this study, sublot sizes are variable, each job has a specific maximum number of sublots and preemption of a job is not permitted. She proposed a procedure which comprises of the steps of determining integer transfer batch sizes for each job and then sequencing the jobs according to the Johnson's rule using the *run-in* and *run-out* times. However, as shown by other authors, solving these subproblems independently does not guarantee an optimal solution.

Vickson [Vic95] treated multiple job lot streaming in a two-machine flow shop where the jobs had sequence-independent setup times which can be either detached or attached (to the first sublot) (F, 2 | P, II, DS/AS | n, C, R | C_{max}). He also included finite transfer times for moving the sublots. He later relaxed his no-job preemption restriction on the model in order to provide global optimality.

Baker [Bak95] studied the multiple job flow shop lot streaming problem using its resemblance to the scheduling problem with arbitrary time lags (F, 2 | P, II, DS/AS | n, E, I | C_{max}). He also showed how to update the recursive formulas to obtain the sublot sizes when the attached or detached setups are added in the model. In his model, preemption of jobs is not allowed.

CHAPTER 3
SOLUTIONS OF PRIMARY LOT STREAMING PROBLEMS

In this chapter, some single job basic lot streaming problems that have been important in the development of the general size basic lot streaming problem will be reviewed in detail. Multi-product lot streaming problems will not be considered here, due to the fact that the assumptions on multi-product problems vary largely across all the models studied so far in the literature and there has not been a standard approach that might yield to further research on the particular model. In this chapter, first, the linear programming (LP) formulation that was developed by Baker [Bak87] for m machine n consistent sublot single-job lot streaming problem ($F, m \mid S, II \mid n, C, R \mid C_{max}$) will be given. Afterwards, the solution techniques that have been developed for some limited size problems will be discussed. Finally, two efficient algorithms that were developed by the author to solve the general size problem will also be presented in this chapter.

3.1 LP Formulations of the $F, m \mid S, II \mid n, C, R \mid C_{max}$ Problem

Baker's LP formulation [Bak87] searches the appropriate sublot sizes for n consistent sublots of a single production lot with an objective of minimizing the makespan defined as the completion time of the last sublot on the last machine in a flow shop environment. In this formulation, an idle period between the processing of two consecutive sublots on the same machine is allowed. As pointed out in the literature survey, when sublot sizes are allowed to vary throughout the shop, the makespan is

shorter than the one obtained under the consistent sublot assumption. However, due to the additional bookkeeping requirements of variable sublot size assumption, the makespan improvement through the consistent sublots may be satisfactory for many manufacturers.

The LP formulation which served as the fundamental of several solution algorithms for lot streaming problems is presented below. First, let n be the number of sublots into which the production lot will be split on each machine and m be the number of machines in the particular flow shop. Furthermore, let T_{ij} denote the completion time of sublot j on machine i. Additionally, since it was assumed that the sublot sizes can be normalized, the fractions allocated to particular sublots are used as the decision variables in the formulation, where x_j represents the fraction associated with sublot j. Lastly, let p_i denote the unit processing time on machine i. Then, the linear programming formulation of the m machine n sublot lot streaming problem is as follows,

(P3.1) minimize T_{mn}

subject to

$$x_1 + x_2 + \ldots + x_n = 1 \tag{3.1}$$

$$T_{11} \geq p_1 x_1 \tag{3.2}$$

$$T_{ij} \geq T_{i,j-1} + p_i x_j, \quad 1 \leq i \leq m,\ 2 \leq j \leq n \tag{3.3}$$

$$T_{ij} \geq T_{i-1,j} + p_i x_j, \quad 2 \leq i \leq m,\ 1 \leq j \leq n \tag{3.4}$$

$$T_{ij} \geq 0, \quad 1 \leq i \leq m,\ 1 \leq j \leq n \tag{3.5}$$

$$x_j \geq 0, \quad 1 \leq j \leq n \tag{3.6}$$

In the above formulation, the constraint (3.1) guarantees that the sum of all sublot sizes is equal to the size of the production lot, which is 1 unit due to normalization. The constraint (3.2) ensures that the first sublot cannot leave the first machine before the

whole sublot is processed. The constraint set (3.3) prevents sublot j from beginning its processing on machine i before sublot j-1 completes its processing on the same machine. This set of constraints is also known as the capacity constraints. The set of constraints shown by (3.4) specifies that sublot j cannot begin its processing on machine i before it completes its processing on machine i-1. This set of constraints is also known as the production constraints. The sets of constraints defined by (3.5) and (3.6) prevent the decision variables from assuming negative values.

The above formulation has ($2mn$ - m - n + 2) constraints excluding the nonnegativity constraints and $n(m + 1)$ decision variables. When m and n are small, the lot streaming problem can be easily solved as a linear program. However, for large values of m and n, the linear programming solution approach to the lot streaming problem will require high capacity LP solvers and high speed computing platforms. Therefore, some efficient alternatives to the linear programming approach may be preferred by many practitioners.

Baker [Bak87] generated a second linear programming formulation nearly half the size of the first one. This alternative formulation focuses on the idle periods that occur in the schedule. Let z_{ij} be the idle period on machine i immediately preceding the processing of sublot j. In order to define the idle periods in terms of the completion times, Baker uses the equation $z_{ij} = \max(0, T_{i-1,j} - T_{i,j-1})$, $1 \le i \le m$, $1 \le j \le n$. There is no need for an idle period on the first machine and therefore $z_{1j} = 0$ for $j = 1,\ldots, n$. The first sublot will not be delayed before its processing on any machine since setup and transportation times are not included in the model. Therefore, the idle period on machine i before processing the first sublot is $z_{i1} = x_1(p_1 + p_2 + \ldots + p_{i-1})$ for $i \ge 2$. In the new formulation, the makespan is

defined by $T_{mn} = z_{m1} + p_m x_1 + \ldots + z_{mn} + p_m x_n$, which is the sum of the idle times and the processing times on the last machine. Other completion time variables are also replaced with the ones in terms of idle periods. Then, Baker's alternative LP formulation is the following.

(P3.2) min $z_{m1} + z_{m2} + \ldots + z_{mn}$

s.t. $$\sum_{h=1}^{n} x_h = 1 \qquad (3.7)$$

$$\sum_{h=1}^{j} (z_{ih} - z_{i-1,h}) + (p_i - p_{i-1}) \sum_{h=1}^{j-1} x_h - p_{i-1} x_j \geq 0, \; 2 \leq i \leq m, \; 1 \leq j \leq n \qquad (3.8)$$

$$z_{ij} \geq 0, \qquad 1 \leq i \leq m, \; 1 \leq j \leq n \qquad (3.9)$$

$$x_j \geq 0, \qquad 1 \leq j \leq n \qquad (3.10)$$

In the objective function, the term $\sum_{h=1}^{n} p_m x_h$, the sum of the processing times on the last machine, is left out since it is a constant. The constraint (3.7) again ensures that the sum of the sublot sizes is equal to 1, which is the size of the production lot in the normalized case. The constraint set (3.8) defines both the capacity and the production constraints. Finally, (3.9) and (3.10) are the nonnegativity constraints for the decision variables. The above formulation has (mn - n + 1) constraints excluding the nonnegativity constraints and mn decision variables. Therefore, sometimes (P3.2) may be more convenient to use than (P3.1), even though its decision variables are not used often in the scheduling.

3.2 Solution of the $F, 2 \mid S, II \mid n, C, R \mid C_{max}$ Problem

Baker [Bak87] solved the two machine problem after developing the linear programming formulations for the general problem. He obtained this solution through the second formulation adapted to the two machine problem as follows.

(P3.3) min $z_1 + z_2 + \ldots + z_n$

s.t. $$\sum_{h=1}^{n} x_h = 1 \qquad (3.11)$$

$$\sum_{h=1}^{j} z_h + (p-1)\sum_{h=2}^{j} x_{h-1} - x_j \geq 0, \quad \text{for } 1 \leq j \leq n \qquad (3.12)$$

$$z_j \geq 0, \qquad \text{for } 1 \leq j \leq n \qquad (3.13)$$

$$x_j \geq 0, \qquad \text{for } 1 \leq j \leq n \qquad (3.14)$$

In the above adaptation of (P3.2), the processing times are re-scaled so that $p_1 = 1$ and $p_2 = p$, since the ratio, $p = p_2 / p_1$, becomes a helpful shortcut in defining the optimal solution. Since the idle periods occur only on the second machine, the first subscript on the idle period variables z_{ij} is dropped from the formulation. Baker showed that the optimal makespan for the lot streaming problem with two machines and an arbitrary number of sublots (n) is the following,

$$M^* = (1 + p + \ldots + p^n) / (1 + p + \ldots + p^{n-1}) \qquad (3.15)$$

since (P3.3) results in

$$z_1^* = 1 / (1 + p + \ldots + p^{n-1}), \text{ and } z_2^* = z_3^* = \ldots = z_n^* = 0 \qquad (3.16)$$

while the optimal sublot sizes are

$$x_j = p^{j-1} / (1 + p + \ldots + p^{n-1}), \quad j = 1,2,\ldots,n, \qquad (3.17)$$

where n is the number of sublots and $p = p_2 / p_1$

In order to prove the optimality of the proposed solution, Baker first showed that the solution given above is a feasible solution to (P3.3). Then he generated the following dual problem corresponding to (P3.3),

(D3.3) max w

s.t. $w - u_j + (p-1)(u_{j+1}+...+u_n) \le 0, \quad \text{for } 1 \le j \le n$ (3.18)

$u_j+...+u_n \le 1, \quad \text{for } 1 \le j \le n$ (3.19)

where dual variable w corresponds to the first constraint of (P3.3) and variable u_j corresponds to the idle period constraint on z_j. Baker obtained the following feasible solution to the dual problem,

$$u_j = p^{n-j} / (1 + p+...+p^{n-1}), \text{ for } 1 \le j \le n \quad (3.20)$$

$$w = 1 / (1 + p+...+p^{n-1}) . \quad (3.21)$$

Here, it can be seen that feasible solutions to the respective problems generate identical objective function values, z_1 in the primal problem and w in the dual problem, while both feasible solutions satisfy the constraints of the corresponding problem in strict equalities. Therefore, the complementary slackness conditions of LP hold and the solutions are optimal in their respective problems [Bak87]. Baker also observed some interesting characteristics of the optimal solution. First, he pointed out that the work allocation among the sublots is uneven; particularly if the first machine is faster than the second, then the sublot sizes are in the increasing order and, if the second machine is faster than the first, then the sublot sizes are in the decreasing order. Second, the makespan gets shorter as the number of sublots increases. Therefore, he conjectured that the ideal number of sublots is infinite and it may be desirable to create as many sublots as

possible. However, it is worthwhile remembering the above considered model works under the assumption of normalized production lot and does not require integer sublot sizes. If that was the case, the number of the sublots would be limited to the number of items in the production lot similar to what is usually done in JIT.

As outlined previously, Trietsch [Tri87] also generated the same solution scheme for the two machine problem. However, his model is somewhat different; he does not normalize sublot sizes and imposes the additional requirement that each sublot be not less than 1. Then, based on the observation that there would be no idle period between the processing of consecutive sublots on the second machine when the setups and transportation times are excluded, he determined that the processing time of a sublot on the first machine must be equal to the processing time of the preceding sublot on the second machine, that is

$$x_j p_1 = x_{j-1} p_2, \qquad 2 \le j \le n \tag{3.22}$$

and from $p = p_2 / p_1$

$$x_j = x_{j-1} p = x_1 p^{j-1}, \quad 2 \le j \le n. \tag{3.23}$$

Since the sum of the individual sublot sizes must add up to the size of the production lot, that is, $x_1 + x_2 + ... + x_n = Q$, the optimal sublot sizes are obtained through the following equations,

$$x_j = \begin{cases} Qp^{j-1}(1-p)/(1-p^n), & p \neq 1 \\ Q/n, & p = 1. \end{cases} \tag{3.24}$$

Trietsch [Tri87] also determined the possible maximum reduction in the makespan, which can be achieved when there are as many sublots as the number of items

in the production lot and each sublot contains only one item. Therefore, he showed that the maximum possible gain (MPG), while satisfying the integrality constraints, would be

$$\text{MPG} = (Q - 1) \min\{p_1, p_2\}, \tag{3.25}$$

where Q is the size of the production lot and p_i, $i = 1, 2$, is the unit processing time on machine i. He also obtained a formula for the marginal makespan reduction obtained by introducing an additional sublot in the model, then he showed that when there are two sublots in the model, more than half of the MPG has already been acquired. Since the marginal contribution of each additional sublot decreases monotonously, Trietsch [Tri87] showed that the maximum sublot number that would achieve the MPG was in the level of $O(\log(Q))$, which contradicts the insight given by Baker's model where sublot number does not have an upper limit [Bak87]. Trietsch also developed some rounding methods in order to obtain the integer sublot size. He stated that assuming the production lot size is large enough, these rounding schemes provide satisfactory results without referring to integer programming approach.

3.3 Solution of the $F, m \mid S, II \mid 2, C, R \mid C_{max}$ Problem

3.3.1 Baker and Pyke's Algorithm

There has been several solution approaches to the m machine two sublot lot streaming problem in a flow shop. First, Baker [Bak87] used his alternative LP formulation (P3.2) which minimizes the idle periods on the last machine in order to solve the problem. He eliminated the second sublot size, x_2, in the formulation via the constraint $x_1 + x_2 = 1$ and obtained the following formula for the optimal size of the first sublot,

$$x_1 = (p_k + \ldots + p_{k+i}) / [(p_k + p_{k+1}) + \ldots + (p_{k+i} + p_{k+i+1})], \quad (3.26)$$

where $1 \le k \le m - 1$ and $0 \le i \le m - k - 1$. However, he has not been able to suggest an effective search method for finding the machine indices k and i.

Baker and Pyke [Bak90] generated a network representation for the lot streaming problem using Baker's LP formulation (P3.1) [Bak87]. They used the capacity and production constraints in the model as the horizontal and vertical precedence relationships of the activity-on-node network shown in Figure 3.1. In this network, node (i, j) represents the processing of sublot j on machine i, which takes $p_i x_j$ time units. The makespan corresponds to the length of the longest path in the network. Since the sublot sizes are decision variables, this network has variable-length activity times. Therefore, the problem is transformed to minimization of the length of the longest path in the network.

Baker and Pyke [Bak90] applied their network representation on two sublot problem. Since specifying x_1 automatically determines x_2, due to $x_2 = 1 - x_1$, they tried to minimize the length of the longest path in the network over all x_1, or simply x, values. They noticed that a path in the network that emanates from (1, 1) follows a route such as (1, 1) - (2, 1) - … - (i-1, 1) - (i, 1) - (i, 2) - (i+1, 2) - … - (m, 2), $1 \le i \le m$, until it reaches to the final node (m, 2) and thus, that path can be represented by machine index i, as shown in Figure 3.2. Here, i is the *crossover machine index*, where the path crosses from the first column to the second. Hence, the length of the path that crosses over machine i, given x is known, is defined as

$$M_i(x) = T(1,i)x + T(i,m)(1-x) = [T(1,i-1) - T(i+1,m)]x + T(i,m) \quad (3.27)$$

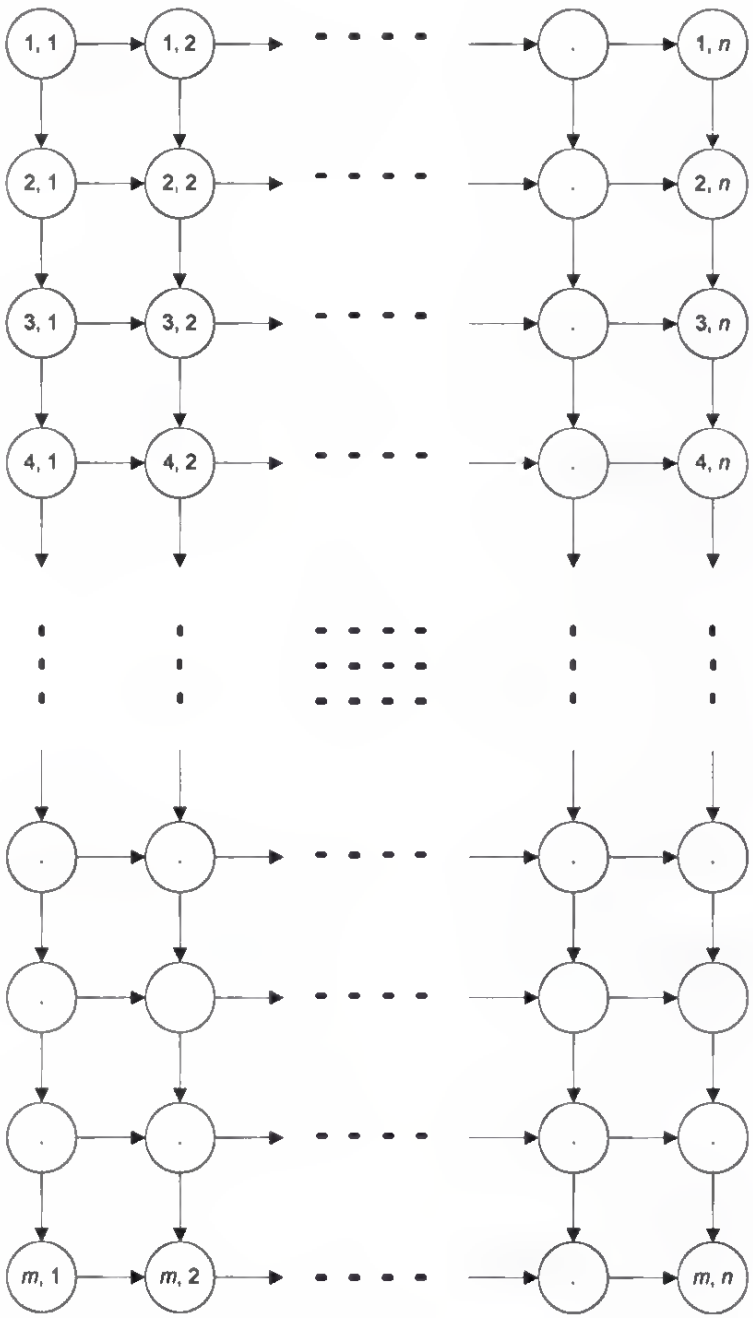

Figure 3.1 Network representation of the *m* machine *n* sublot problem

where $T(j,k) = \sum_{h=j}^{k} p_h$. Therefore, the length of the longest path or the *critical path*, for the given x value, is obtained from

$$M(x) = \max_{1 \le i \le m} \{[T(1,i-1) - T(i+1,m)]x + T(i,m)\} . \tag{3.28}$$

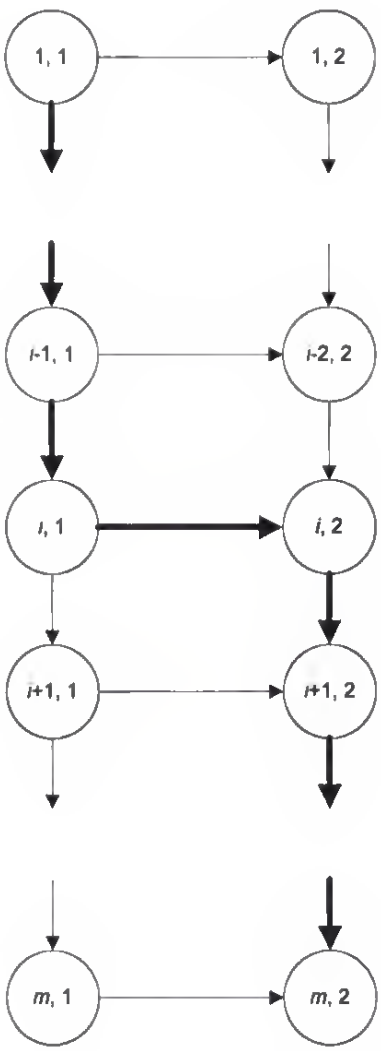

Figure 3.2 Network representation of the m machine 2 sublot problem

Baker and Pyke [Bak90] observed that a critical path that crosses over at machine i in the network stays critical for a range of x values, as characterized below,

$$x \le \frac{T(i, j-1)}{T(i, j-1) + T(i+1, j)}, \text{ for } j > i, \tag{3.29}$$

and

$$x \ge \frac{T(j, i-1)}{T(j, i-1) + T(j+1, i)}, \text{ for } j < i. \tag{3.30}$$

They noticed that in order for a machine to be a critical index, the lower and upper limits associated with that machine must be consistent. Baker and Pyke redefined the lower and upper limits as follows,

$$x_{1i} = \min_{j>i} \left\{ \frac{T(i, j-1)}{T(i, j-1) + T(i+1, j)} \right\}, \tag{3.31}$$

and

$$x_{2i} = \max_{j<i} \left\{ \frac{T(j, i-1)}{T(j, i-1) + T(j+1, i)} \right\}, \tag{3.32}$$

and reiterated that if $x_{2i} > x_{1i}$, machine i cannot be critical and thus, it must be skipped in the enumeration process.

Baker and Pyke determined that for a critical machine index i, the size of the first sublot, x, must assume either of the two limits, depending on the sign of the coefficient of x in the objective function in (3.27). If the coefficient is negative, that is, $(T(1, i-1) - T(i+1, m) < 0)$, the optimal choice of x will be given by the tightest condition in (3.29) in order to make it as large as possible, that is, $x = x_{1i}$. If the coefficient is positive, $(T(1, i-1) - T(i+1, m) > 0)$, x will be minimized by the tightest condition in (3.30), that is $x = x_{2i}$. Finally, the optimal makespan will be the following,

$$M^* = \min_{0 \le x \le 1} \{ M(x) | x_{2j} \le x_{1j} \}. \tag{3.33}$$

Baker and Pyke also observed that due to the definition of x_{1j} and x_{2j}, the optimal makespan is also obtained via index j, that is, there will be at least two critical paths. Therefore, in the optimal schedule there will be two machines on which there can be no idle period. Then, they proceeded to develop a fast solution scheme where they assumed that the slowest machine is always a critical machine. Depending on this unstable assertion, they search a second critical index in order to determine the size of the first sublot. However, as shown by Williams and Tüfekçi, the slowest machine may not always be a critical index [Wil92]. Therefore, Baker and Pyke's algorithm, which has a computational complexity of $O(m)$, often results in suboptimal results. This would have been avoided simply by checking all possible pairs of critical indices, which leads to $O(m^2)$ complexity level.

3.3.2 Williams and Tüfekçi's $O(m^2)$ Algorithm

Williams and Tüfekçi [Wil92] proved that Baker and Pyke's algorithm does not always yield optimal results since the path defined by the slowest machine may not necessarily be the critical path in the network. Afterwards, they developed several algorithms that find optimal solutions.

Williams and Tüfekçi [Wil92] first developed a complete enumeration method that checks every machine index on the network. This algorithm visits each machine and determines the length of the critical path corresponding to the particular machine. The complexity of this algorithm $O(m^2)$, since it checks every machine and requires a second machine index for sublot size computation.

In the same work, Williams and Tüfekçi [Wil92] developed a second algorithm which is a bisection-based algorithm using Baker's alternative LP formulation where idle periods on each machine are used as decision variables. They observed that the formulation for the two sublot problem is ready to be represented by a directed path network. Due to the special structure of the network, given the information on the processing times and the size of the first sublot, they have been able to determine the idle time that occurs on the last machine before the second sublot starts its processing in $O(m)$ time, where m is the number of machines in the problem. Therefore, in order to provide the network algorithm with the trial sublot size, they used a bisection framework, and they repeated the network algorithm for feedback on how to reduce the domain. After K steps, the Bisection Algorithm will come within $2^{-K} + 2\varepsilon\ (1-2^{-K})$ of the optimal sublot size, where 2ε is the width of the bisection interval. Total complexity of this algorithm is $O(Km)$.

Williams and Tüfekçi [Wil92] developed yet another optimal algorithm for the two sublot problem which also runs in $O(m^2)$ time but practically faster than the complete enumeration. They used the project network representation and determined the important properties of the critical paths in the network. Before presenting their algorithm, it is worthwhile to review some of their definitions and lemmas. The definitions that introduce the concept of feasible indices, which are candidates to be critical indices, are as follows,

Definition 3.1. *Machine index p is a feasible index if there exists an x_p which satisfies one of the following inequalities,*

$$\frac{T(i,p-1)}{T(i,p-1)+T(i+1,p)} < x_p \le \frac{T(p,j-1)}{T(p,j-1)+T(p+1,j)}, \quad \forall i < p \text{ and } \forall j > p \tag{3.34}$$

where i, p and j are machine indices and $c_p \leq 0$, *or*

$$\frac{T(i,p-1)}{T(i,p-1)+T(i+1,p)} \leq x_p < \frac{T(p,j-1)}{T(p,j-1)+T(p+1,j)}, \quad \forall i < p \text{ and } \forall j > p \tag{3.35}$$

where i, p and j are machine indices and $c_p \geq 0$. *Here* $c_p = T(1,p-1) - T(p+1,m)$ *and m is the index of the last machine.*

Definition 3.2. *The ratio induced by a feasible index p is*

$$\min_{j>p}\left\{\frac{T(p,j-1)}{T(p,j-1)+T(p+1,j)}\right\} = \frac{T(p,q-1)}{T(p,q-1)+T(p+1,q)} \tag{3.36}$$

for $c_p \leq 0$ *where q,* $q > p$, *is the largest machine index which satisfies the equation above, or*

$$\max_{i<p}\left\{\frac{T(i,p-1)}{T(i,p-1)+T(i+1,p)}\right\} = \frac{T(q,p-1)}{T(q,p-1)+T(q+1,p)} \tag{3.37}$$

for $c_p \geq 0$ *where q,* $q < p$, *is the smallest machine index which satisfies the equation above.*

Definition 3.3. *Machine p points to machine q, or* $M_p \rightarrow M_q$, *if*

$$x_p = \frac{T(p,q-1)}{T(p,q-1)+T(p+1,q)} \tag{3.38}$$

or

$$x_p = \frac{T(q,p-1)}{T(q,p-1)+T(q+1,p)} \tag{3.39}$$

where x_p *is a ratio induced by a feasible index p.*

The lemmas that Williams and Tüfekçi used to develop the algorithm are the following.

Lemma 3.1. *A critical machine p exists such that index p is a feasible index.*

Lemma 3.2. *If index p is a feasible index such that $M_p \rightarrow M_q$, then index q is a feasible index.*

Lemma 3.3. *Let index p be a feasible index such that $M_p \rightarrow M_q$, then no index j, such that $p < j < q$ for $p < q$, or $q < j < p$ for $p > q$ can be a feasible index.*

Lemmas 3.1-3 provide a systematic way of searching for a critical machine. Lemma 3.1 states that a critical machine whose index is also a feasible index exists in every lot streaming problem. Next, Lemma 3.2 ensures that already having a feasible index, another one can be found. Therefore, the first two lemmas are helpful in obtaining a sequence of feasible indices, given an initial feasible index. However, even if a sequence of feasible indices is obtained, it is necessary to know if any feasible indices have been excluded between the initial feasible index and the last index in the generated sequence. Lemma 3.3 guarantees that all feasible indices will be included in the generated sequence. Furthermore it also states that, once a feasible index is found, no index between the feasible index that has been just found and the feasible index it points to can be a feasible index.

Lemma 3.4. *If index p is a feasible index such that $M_p \rightarrow M_q$ and $c_p c_q \leq 0$, then $M_q \rightarrow M_p$.*

Lemma 3.5. *Let index p be a feasible index such that $M_p \rightarrow M_q$ and $c_p c_q \leq 0$ if and only if machine p is a critical machine.*

Lemma 3.4 and Lemma 3.5 state the necessary and sufficient conditions for a feasible index to be a critical index. Below, Lemma 3.6 ensures that there will be no cycling in the search for a critical feasible index. Essentially, it states that the only pair of feasible indices pointing each other is the pair where one of the indices is a critical index, which also leads to the criticality of the index that is pointed by a critical index.

Lemma 3.6. *If machine p is a critical machine whose index is a feasible index such that* $M_p \rightarrow M_q$, *then no indices s and t,* $s \neq p$, $s \neq q$, $t \neq p$ *and* $t \neq q$ *exist such that s and t are feasible indices with* $M_s \rightarrow M_t$, *and* $M_t \rightarrow M_s$.

Lemma 3.7. *If machine p is the machine with the largest processing time such that* $c_p \leq 0$, $t_i < t_p$, $\forall\ i < p$ *and* $t_j \leq t_p$, $\forall\ j > p$, *or* $c_p \geq 0$, $t_j < t_p$, $\forall\ j > p$ *and* $t_i \leq t_p$, $\forall\ i < p$, *then index p is a feasible index.*

Lemma 3.8. *Let machine p be the machine with the largest processing time such that its index is a feasible index, then if* $c_p < 0$, *no index i,* $i < p$, *can be a feasible critical index or if* $c_p > 0$, *no index i,* $i > p$, *can be a feasible critical index.*

Lemma 3.7 and Lemma 3.8 provide that the slowest machine in the flow shop is also a feasible index and it can be used as an initial feasible index in the search of a critical index without leaving any potential critical index out of the evaluation.

After the necessary groundwork, Williams-Tüfekçi algorithm for *m* machine 2 sublot lot streaming problem is as follows.

Algorithm D

1. Find an index *i* such that

$$t_i > t_j \ \forall\ j < i$$

$$t_i \geq t_j \ \forall\ j > i$$

for $c_i \leq 0$, or

$$t_i > t_j \ \forall\ j > i$$

$$t_i \geq t_j \ \forall\ j < i$$

for $c_i > 0$, where $c_i = T(1, i - 1) - T(i + 1, m)$. Index *i* is the initial feasible index.

2a. If $c_i \leq 0$, then

$$z = \min_{j>i}\left\{\frac{T(i,j-1)}{T(i+1,j)}\right\} = \frac{T(i,p-1)}{T(i+1,p)}$$

where p, $p > i$, is the largest index for which the above equation is true. Hence, index p is a also feasible index. Calculate c_p. If $c_p > 0$, then go to Step 3, otherwise set $i = p$ and repeat Step 2a.

2b. If $c_i > 0$, then

$$z = \max_{j<i}\left\{\frac{T(j,i-1)}{T(j+1,i)}\right\} = \frac{T(p,i-1)}{T(p+1,i)}$$

where p, $p < i$, is the smallest index for which the above equation is true. Hence, index p is a also feasible index. Calculate c_p. If $c_p \leq 0$, then go to Step 3, otherwise set $i = p$ and repeat Step 2b.

3. Let $x = \frac{z}{z+1}$. The optimal makespan is $M = c_i x + T(i,m)$.

Williams and Tüfekçi also proved that the above algorithm provides optimal solutions in $O(m^2)$ time complexity and stated that even though its complexity is equal to that of the complete enumeration, the above algorithm often results in fewer steps.

3.3.3 Williams and Tüfekçi's Linear-Time Algorithm

Upon developing an $O(m^2)$-time optimal algorithm presented in the previous section, Williams and Tüfekçi's research [Wil95] resulted in a linear-time optimal algorithm for the m machine two sublot problem. They used the results of Megiddo [Meg83] for the 2-variable m-constraint LP problem in the development of this algorithm.

In the network representation for the m machine two sublot problem, shown in Figure 3.2, the length of the longest path for a given set of sublot sizes determine the makespan. Since the nodes of the network have variable weights, the minimum makespan is obtained through minimizing the longest path possible in the particular network. Here, for the given values of x_1 and x_2, size of the first and second sublot, respectively, any possible path between (1, 1) and (m, 2) is represented by $M_i(x_1, x_2)$, where i is the machine index where the path switches from the first column to the second column in the network and its length is defined as,

$$M_i(x_1, x_2) = T(1,i)x_1 + T(i,m)x_2, \tag{3.40}$$

where $T(i,j) = \sum_{k=i}^{j} p_k$.

Since the sum of the two sublot sizes must be equal to the size of the production lot, that is, $x_1 + x_2 = 1$, x_2 can be replaced by $1 - x_1$, and simply x can be used instead of x_1. Then, the length of the path becomes a linear function of x as follows,

$$M_i(x) = T(1,i)x + T(i,m)(1-x),$$

by arranging the terms,

$$M_i(x) = \big(T(1,i-1) - T(i+1,m)\big)x + T(i,m),$$

and by defining $c_i = T(1,i-1) - T(i+1,m)$,

$$M_i(x) = c_i x + T(i,m), \quad \text{for } i = 1,\ldots,m. \tag{3.41}$$

Therefore, the path length for the specified sublot combination is a line in $\mathcal{R}^2$, defined in the domain $0 \le x \le 1$. The makespan for a given size of the first sublot, $M(x)$, will be the length of the longest path in the network, and it will be defined as,

$$M(x) = \max_{i=1,\ldots,m} M_i(x). \tag{3.42}$$

Hence, the makespan of the m machine two sublot problem is the upper envelope of the intersection of the lines $c_i x + T(i,m)$, $i = 1,\ldots,m$, in $\mathcal{R}^2$, and it is a convex function. Since the objective in this problem is to obtain the minimum makespan and the sublot sizes that yield that value, correspondingly, the longest path must be minimized in the network representation over the feasible domain of the first sublot. Then, the optimal makespan, M^*, is represented by

$$M^* = \min_{0 \le x \le 1} \max_{i=1,\ldots,m} M_i(x). \tag{3.43}$$

Figure 3.3 below depicts the objective function of the min-max problem over the feasible region where the function is defined. In this graph, each linear function represents the length of a path associated with a crossover machine index. The maximum of all linear functions, the convex envelope marked by a dark line, corresponds to the longest path in the network, that is the makespan for any specific value of the first sublot size. Finally, the optimum makespan is located where the convex envelope takes its lowest value over the feasible region.

The above min-max problem can also be represented in the form of a linear program with two decision variables and m constraints as follows,

(P3.4) minimize M

subject to $M - c_i x \ge T(i,m), \quad i = 1,\ldots,m$ (3.44)

$M \ge 0$ (3.45)

$0 \le x \le 1,$ (3.46)

where x is the size of the first sublot and M is the makespan of the problem. The above formulation contains only two decision variables, x and M. Also, without considering the nonnegativity constraints and upper bound on x, it has m constraints.

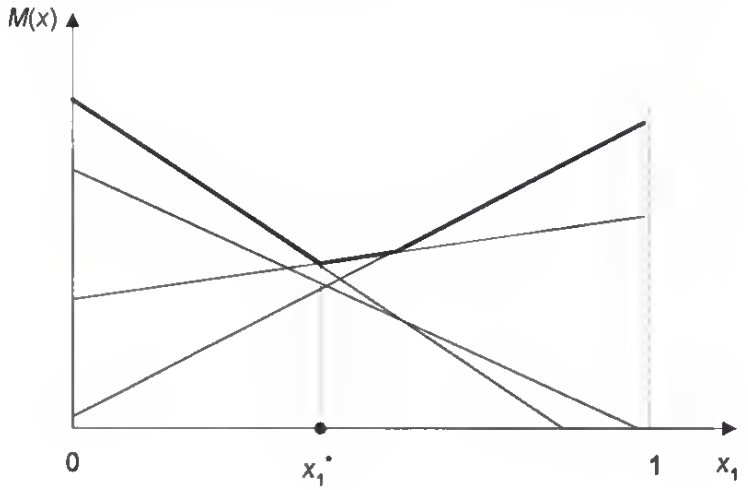

Figure 3.3 Representation of the m machine 2 sublot problem in $\mathfrak{R}^2$

Megiddo [Meg83] showed that a linear program containing two decision variables and m constraints could be solved in $O(m)$ operations. Williams and Tüfekçi [Wil95], stating that the m machine two sublot problem is also solvable in $O(m)$ operations, modified the solution procedure of Megiddo to solve the particular problem. Their algorithm is presented below. First, they define $NUM(I)$ as the number of elements in the set I. In what follows, $\lfloor x \rfloor$ represents the largest integer smaller than or equal to x.

Megiddo m x 2

0. Let $I = \{i_1, i_2, \ldots, i_m\}$ be the set of machine indices in the lot streaming problem. If $m = 2$, go to Step 7, else set $x_L = 0$ and $x_H = 1$.

1. For each $j \in J = \left\{1,3,\ldots,2\left\lfloor \frac{NUM(I)}{2} \right\rfloor - 1\right\}$ set

$$x_{J,J+1} = \frac{T(i_{J+1},m) - T(i_J,m)}{c_{i_J} - c_{i_{J+1}}}$$

2. For each $j \in J$, if $x_{JJ+1} \geq x_H$, then update J and I as $J \setminus \{j\}$ and $I \setminus \{k\}$, respectively, where

$$k = \begin{cases} i_J & \text{if } c_{i_J} > c_{i_{J+1}} \\ i_{J+1} & \text{otherwise.} \end{cases}$$

else if $x_{JJ+1} \leq x_L$, then update then update J and I as $J \setminus \{j\}$ and $I \setminus \{k\}$, respectively, where

$$k = \begin{cases} i_J & \text{if } c_{i_J} < c_{i_{J+1}} \\ i_{J+1} & \text{otherwise.} \end{cases}$$

3. If $NUM(I) = 2$, go to Step 7,

else if $J = \{\varnothing\}$, relabel $I = \{i_1, i_2, \ldots, i_{NUM(I)}\}$ and go to Step 1,

else set

$$\bar{x} = \sum_{j \in J} \frac{x_{J,J+1}}{NUM(J)}$$

4. Let $K = \{k \in I : c_k \bar{x} + T(k,m) \geq c_i \bar{x} + T(i,m), \forall i \in I\}$.

Set $slope1 = \min_{k \in K}\{c_k\}$ and $slope2 = \max_{k \in K}\{c_k\}$.

5. If $(slope1)(slope2) \leq 0$, then set $x^* = \bar{x}$

and set $M^* = \max_{i \in I}\{c_i \bar{x} + T(i,m)\}$, stop.

6. If $slope1 > 0$, then set $x_H = \bar{x}$ and for each $j \in J$ such that $x_{J,J+1} \geq \bar{x}$,

update I as $I \setminus \{k\}$ where

$$k = \begin{cases} i_j & \text{if } c_{i_j} > c_{i_{j+1}} \\ i_{j+1} & \text{otherwise.} \end{cases}$$

If $slope1 < 0$, then set $x_L = \bar{x}$ and for each $j \in J$ such that, $x_{j,j+1} \leq \bar{x}$

update I as $I \setminus \{k\}$ where

$$k = \begin{cases} i_j & \text{if } c_{i_j} < c_{i_{j+1}} \\ i_{j+1} & \text{otherwise.} \end{cases}$$

7. Relabel $I = \{i_1, i_2, \ldots, i_{NUM(I)}\}$. If $NUM(I) = 2$, the optimal size of sublot 1 is

$$x^* = \frac{T(i_2, m) - T(i_1, m)}{c_{i_1} - c_{i_2}}, \text{ go to Step 8,}$$

else go to Step 1.

8. The optimal makespan is $M^* = c_{i_1} x^* + T(i_1, m)$.

Megiddo [Meg83] showed that the above procedure results at most $O(m)$ operations. Particularly, when $m \leq 3$, only one application of the steps 1-7 of the algorithm solves the problem, while when $m = 4$, at most two iterations containing the steps 1-7 are needed to solve the m machine two sublot problem.

3.4 Solution of the $F, 3 \mid S, II \mid n, C, R \mid C_{max}$ Problem

Baker [Bak87] tried to solve the three machine problem with only two sublots allowed. He used his alternative LP formulation where the idle periods are the decision variables. He observed that the ratio of the two sublot sizes varies depending on a comparison among the processing times on all of the machines. Baker's limited findings on the three machine problem are presented below.

If $p_2^2 > p_1 p_3$ and $p_1 > p_3$,

$$x_1 = p_1/(p_1 + p_2),$$

$$x_2 = p_2/(p_1 + p_2),$$

$$M^* = \frac{p_1^2 + p_2^2 + p_1 p_2 + p_2 p_3}{p_1 + p_2}. \tag{3.47}$$

If $p_2^2 > p_1 p_3$ and $p_1 < p_3$,

$$x_1 = p_2/(p_2 + p_3),$$

$$x_2 = p_3/(p_2 + p_3),$$

$$M^* = \frac{p_2^2 + p_3^2 + p_1 p_2 + p_2 p_3}{p_2 + p_3}. \tag{3.48}$$

If $p_2^2 < p_1 p_3$,

$$x_1 = (p_1 + p_2)/(p_1 + 2p_2 + p_3),$$

$$x_2 = (p_2 + p_3)/(p_1 + 2p_2 + p_3),$$

$$M^* = \frac{p_1^2 + p_2^2 + p_3^2 + 2p_1 p_2 + 2p_2 p_3 + p_1 p_3}{p_1 + 2p_2 + p_3}. \tag{3.49}$$

Glass, Gupta and Potts extended the three machine to include an arbitrary number of sublots [Gla94]. They used the renowned network representation in order to solve the 3-machine n sublot lot streaming problem. They determined that the critical path in the network can take one of the specified forms for which they obtained the sublot sizes and makespan formulas. Before presenting Glass et al.'s algorithms, it will be helpful to review the definitions and the theoretical background of their paper.

First, they showed that every sublot is critical in the project network, that is, the critical path has segments such as (i, j) - $(i+1, j)$ for some machine i, $1 \le i \le m-1$, for every

sublot j, $1 \le j \le n$, and that all sublot sizes are positive, as seen in Theorem 3.1 and its corollary, respectively [Gla94].

Theorem 3.1. *In any network of optimal sublot sizes, every sublot is critical.*

Corollary 3.1. *In any optimal solution, all sublot sizes are positive.*

Glass et al. [Gla94] also defined that machine i, $1 \le i \le m$, is *critical from sublot* j, $1 \le j \le n$-1, if there is a critical segment such as (i, j) - $(i, j+1)$ and they proved that there are at least two critical machines from each sublot j for $1 \le j \le n$-1, as in Theorem 3.2.

Theorem 3.2. *There exists a vector of optimal sublot sizes x for which the project network contains at least two critical machines from sublot j for* $1 \le j \le n$-1.

Glass et al. proved in Theorem 3.3 that the critical path in the network can take one of the specific shapes depending on a comparison between the processing times on the three machines. They used the previous theorems in the proof of Theorem 3.3 below [Gla94].

Theorem 3.3. *In a network of optimal sublot sizes:*

(a) *when* $p_2^2 < p_1 p_3$, *all critical paths are of the form* $(1, 1)$ - … - $(1, j)$ - $(2, j)$ - $(3, j)$ - … - $(3, n)$ *for* $1 \le j \le n$;

(b) *when* $p_2^2 = p_1 p_3$, *all paths in the network are critical;*

(c) *when* $p_2^2 \ge p_1 p_3$, *for some sublot k, all critical paths are of the form* $(1, 1)$ - … - $(1, j)$ - $(2, j)$ - … - $(2, l)$ - $(3, l)$ - … - $(3, n)$ *for* $1 \le j \le k \le l \le n$.

The above theorem helped Glass et al. [Gla94] in proving the following theorems and lemma which provides a method of finding the optimal sublot sizes and makespan.

Theorem 3.4. *For the case* $p_2^2 \le p_1 p_3$, $x^{(0)}$ *is a vector of optimal sublot sizes, where*

$$x_j^{(0)} = q^{j-1} x_1^{(0)} \quad \text{for } 1 \le j \le n,$$

for $q = (p_2 + p_3)/(p_1 + p_2)$, *and* $1/x_1^{(0)} = \begin{cases} (q^n - 1)/(q-1) & \text{if } p_1 \ne p_3, \\ n & \text{if } p_1 = p_3. \end{cases}$

Using the optimal sublot sizes given above, the optimal makespan of the three machine problem will be the following,

$$M(x) = \begin{cases} p_3 + (p_3 - p_1)/(q^n - 1) & \text{if } p_1 \ne p_3, \\ p_3 + (p_1 + p_2)/n & \text{if } p_1 = p_3. \end{cases} \tag{3.50}$$

Theorem 3.5. *For the case* $p_2^2 > p_1 p_3$, $x^{(k)}$, *is a vector of optimal sublot sizes for some* $k \in \{1,\ldots,n\}$, *where*

$$x_j^{(k)} = \begin{cases} q_1^{k-j} x_k^{(k)} & \text{for } 1 \le j \le k, \\ q_3^{j-k} x_k^{(k)} & \text{for } k \le j \le n, \end{cases}$$

for $q_1 = p_1/p_2$, $q_3 = p_3/p_2$, *and*

$$1/x_k^{(k)} = \begin{cases} (q_1^k - 1)/(q_1 - 1) + (q_3^{n-k+1} - 1)/(q_3 - 1) - 1 & \text{if } p_1 \ne p_2,\ p_2 \ne p_3, \\ k - 1 + (q_3^{n-k+1} - 1)/(q_3 - 1) & \text{if } p_1 = p_2,\ p_2 \ne p_3, \\ (q_1^k - 1)/(q_1 - 1) + n - k & \text{if } p_1 \ne p_2,\ p_2 = p_3. \end{cases}$$

Theorem 3.4 above gives the optimal sublot sizes for the case $p_2^2 > p_1 p_3$, assuming the pattern changing sublot, k, is known. Therefore, in order to obtain the values of the optimal sublot sizes, the value of k must be determined. Lemma 3.9 below gives the information on how to determine k.

Lemma 3.9. *When* $p_2^2 > p_1 p_3$, $M(x^{(k)})$ *is a discrete convex function in* k, $M(x^{(k+1)}) - M(x^{(k)})$ *has the same sign as* $\Delta(k)$ *for* $1 \le k \le n - 1$, *where*

$$\Delta(k) = \begin{cases} (\tilde{q}_1^{k+1} - 1)/(\tilde{q}_1 - 1) - (\tilde{q}_3^{n-k+1} - 1)/(\tilde{q}_3 - 1) & \text{if } p_1 \neq p_2,\ p_2 \neq p_3, \\ k + 1 - (\tilde{q}_3^{n-k+1} - 1)/(\tilde{q}_3 - 1) & \text{if } p_1 = p_2,\ p_2 \neq p_3, \\ (\tilde{q}_1^{k+1} - 1)/(\tilde{q}_1 - 1) - (n - k + 1) & \text{if } p_1 \neq p_2,\ p_2 = p_3, \end{cases}$$

where $\tilde{q}_1 = 1/q_1 = p_2/p_1$ and $\tilde{q}_3 = 1/q_3 = p_2/p_3$.

Lemma 3.9 states that makespan of the three machine problem is a convex function over the discrete scale of sublots, which leads to the result that the exact value of the pattern changing sublot, k, can be found by a bisection search as expressed below in Theorem 3.6.

Theorem 3.6. *For* $p_2^2 > p_1 p_3$, *the value of* k^* *is found by bisection search in* $O(\log(n))$ *time.*

As seen from the above statements, the optimal sublot sizes can be found in $O(n)$ time. However, when $p_2^2 > p_1 p_3$, an extra $O(\log(n))$ time is required to find the pattern changing sublot, k, before obtaining the optimal sublot sizes and the makespan.

It is also seen that there is a proportional relationship between the sublot sizes in the three machine problem, although it is more complex than the case in the two machine problem. In the two machine problem, the ratio of x_i / x_{i+1}, $1 \le i \le n$-1, is simply p_1 / p_2. However, this ratio varies depending on the case in the three machine problem. When $p_2^2 \le p_1 p_3$, the ratio of x_i / x_{i+1}, $1 \le i \le n$-1, is $(p_1 + p_2)/(p_2 + p_3)$. If $p_2^2 > p_1 p_3$, then $x_i / x_{i+1} = p_1 / p_2$ for $1 \le j \le k$ and $x_i / x_{i+1} = p_2 / p_3$ $k \le j \le n$, where k is the pattern changing sublot.

3.5 Solution of the $F, m \mid S, II \mid 3, C, R \mid C_{max}$ Problem

The methods that have been developed for the m machine 2 sublot problem were discussed earlier. In this section, the solution procedure that was specifically designed for the m machine 3 sublot problem will be introduced. Although the development of this algorithm is similar to that of the 2 sublot problem in terms of the use of the project network representation, it has not been as straight forward as in the two sublot case where assigning a value to the size of a sublot automatically specifies the size of the other sublot.

Williams, Tüfekçi and Akansel [Wil97] developed an optimal algorithm for the m machine 3 sublot lot streaming problem using the network representation. The project network that represents the m machine 3 sublot problem is shown in Figure 3.4 below. In this network, a node (i, s) represents the processing of sublot s, $s = 1, 2, 3$, on machine i, $i = 1, \ldots, m$. Hence, there are three columns of nodes each representing the processing of a sublot on the entire flow shop. As in the two sublot case, the length of the longest path for a given set of sublot sizes corresponds to the makespan of the lot streaming problem. The optimal makespan is again obtained by determining the set of sublot sizes that yields the minimum longest path.

Williams et al. [Wil97] stated that any path traversing from node (1, 1) to node $(m,3)$ can be represented by two machine indices, i and j, $1 \leq i \leq j \leq m$, where the path crosses over to the second column from the first column at node $(i, 1)$ and crosses over to the third column from the second column at node $(j, 2)$. Therefore, the length of the path

defined by indices i and j, $i \le j$, can be denoted by $M_{ij}(x_1, x_2, x_3)$ and for a given set of the sublot sizes x_1, x_2 and x_3, it is equal to the following,

$$M_{ij}(x_1, x_2, x_3) = T(1,i)x_1 + T(i,j)x_2 + T(j,m)x_3, \tag{3.51}$$

where $T(i,j) = \sum_{k=i}^{j} p_k$. Since the longest path in the network corresponds to the makespan for the specified sublot sizes x_1, x_2 and x_3, it is necessary to determine the path with the maximum length, that is, the makespan for any set of sublot sizes is given by the following,

$$M(x_1, x_2, x_3) = \max_{i \le j}\{M_{ij}(x_1, x_2, x_3)\}. \tag{3.52}$$

The above function is defined in the domain

$$X = \{\mathbf{x} \in R^3 | x_1 + x_2 + x_3 = 1,\ 0 < x_i < 1,\ i = 1,\ 2,\ 3\},$$

since the production lot is distributed among the three sublots. Therefore, the function $M(x_1, x_2, x_3)$ corresponds to the upper envelope created by the intersecting hyperplanes $M_{ij}(x_1, x_2, x_3)$, $i \le j$, in the domain X. Since the domain X is a convex set and the function $M(x_1, x_2, x_3)$ is a convex function defined over a convex set, it is guaranteed that a local minimum solution of the function $M(x_1, x_2, x_3)$ is also a global minimum.

The optimal makespan of the m machine 3 sublot lot streaming problem is expressed as follows,

$$M^* = \min_{\mathbf{x} \in X}\{M(x_1, x_2, x_3)\}. \tag{3.53}$$

Above formulation implies that the optimal solution can also be found by obtaining all the longest paths in the network and then finding the path with the minimal length among all the longest paths and the sublot sizes that yield the longest path with minimal length.

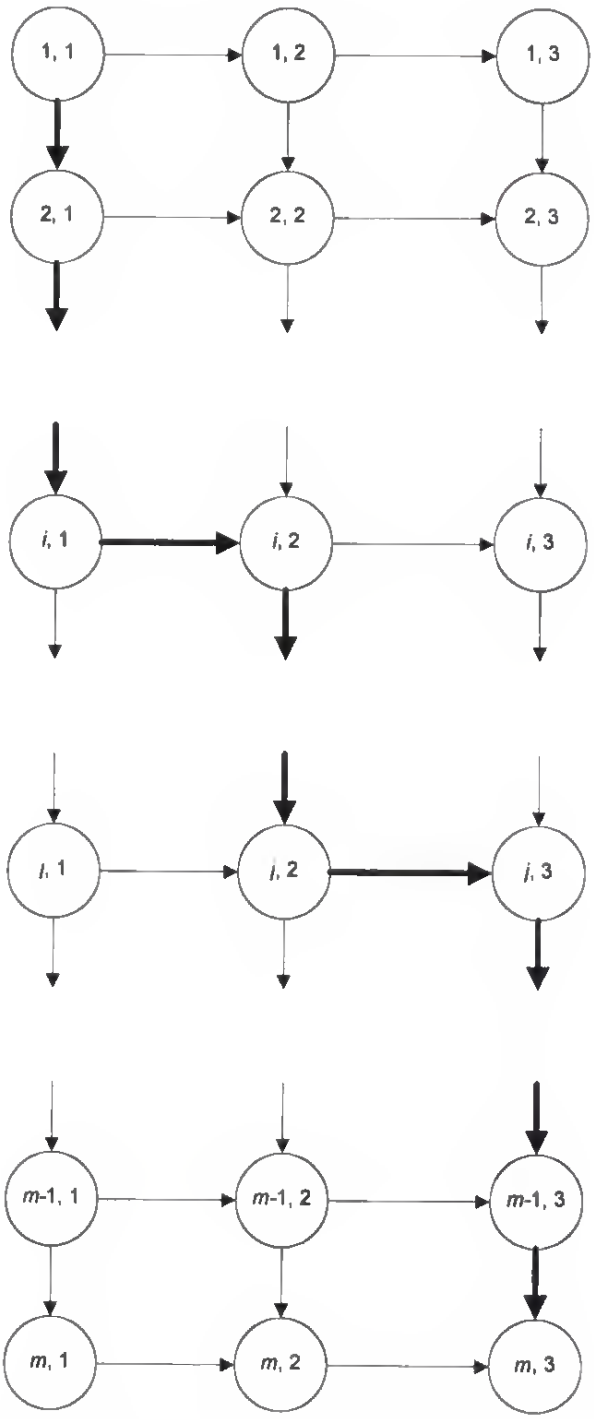

Figure 3.4 Network representation of the m machine 3 sublot problem.

Williams et al. [Wil97] provided some theorems and lemmas supporting the groundwork of their algorithm for the *m* machine 3 sublot lot streaming problem. The theorems and lemmas that are necessary to comprehend the algorithm will be given below. Theorem 3.7 shows that the ratios of the sizes of the consecutive sublots must have some lower and upper bounds in terms of the length of some path segments in the network.

Theorem 3.7. *If the indices i and j, i ≤ j, define a longest path in the network representation of the m machine* 3 *sublot problem for sublot sizes* x_1, x_2, x_3, *the ratio of sublot size* 1 *to sublot size* 2, $z_1 = x_1/x_2$, *must be bounded as*

$$\frac{T(h,i-1)}{T(h+1,i)} \le z_1 \le \frac{T(i,g-1)}{T(i+1,g)}$$

where $1 < i < m$, $h = 1, \ldots, i\text{-}1$ *and* $g = i+1, \ldots, j$, *and the ratio of sublot size* 2 *to sublot size* 3, $z_2 = x_2/x_3$, *must be bounded as*

$$\frac{T(l,j-1)}{T(l+1,j)} \le z_2 \le \frac{T(j,k-1)}{T(j+1,k)}$$

where $i \le j < m$, $l = i, \ldots, j\text{-}1$ *and* $k = j+1, \ldots, m$.

Lemma 3.10 below states that the first and last machine indices are always on all the longest paths.

Lemma 3.10. *Machine index* 1 *or m can always be used to define longest paths in the network representation of the m machine* 3 *sublot lot streaming problem.*

Williams et al. [Wil97] then presented a procedure that will list all of the feasible indices in the problem, which will constitute a very important step in the main algorithm. They stated that the definition and lemmas given in the section where the two sublot

problem is discussed also apply in the 3 sublot problem. The feasible search algorithm is given below,

Feasible Index Search Algorithm

1. Let $I = \{\varnothing\}$ be the initial set of feasible indices.
2. Set $i = 1$, this is the first feasible index. Set $I = I \cup i$.
3. Find the largest index i', such that

$$\min_{i<k}\left\{\frac{T(i,k-1)}{T(i+1,k)}\right\} = \left\{\frac{T(i,i'-1)}{T(i+1,i')}\right\}$$

   Then i' is a feasible index. Set $I = I \cup i'$ and $i = i'$. If $i < m$ repeat this step.
4. I is a set of feasible indices.

Lemma 3.11 below enhances the fundamental of the feasible search algorithm. It states that feasible indices can be obtained from machine index 1 through machine index m as well as from machine index m through machine index 1.

Lemma 3.11. *Let index i be a feasible index. If* $i \xrightarrow{feasible} j$, *then* $j \xrightarrow{feasible} i$, *where* $x \xrightarrow{feasible} y$ *shows that the feasible index y can be obtained through feasible index x.*

Williams et al. [Wil97] provided the lemmas below in order to describe the pattern of the critical path in the network representation of the m machine 3 sublot problem. Lemma 3.12 states that when a longest path traverses from the node (1, 1) to the node (m, 3), the direction of each horizontal arc on the longest path is from sublot j to sublot j+1, $1 \le j < n$, and the direction of each vertical arc on the longest path is from machine index i to machine index i+1, $1 \le i < m$. Lemma 3.13 states that there are some alternative longest paths for the same set of sublot sizes in the network.

Lemma 3.12. *For any given vector* $\mathbf{x} = (x_1, x_2, x_3)$, *the indices of any longest path are nondecreasing both in machine index and in sublot index.*

Lemma 3.13. *If* $\mathbf{x}^*$ *is an optimal solution to an m machine* 3 *sublot problem, then there exists at least two distinct longest paths in the network representation of the problem.*

Following the above lemmas, Williams et al. [Wil97] provided Definition 3.4 to describe the critical path in the network. The following lemmas extend the information available on the pattern of the critical path. For example, Lemma 3.14 states that when there is a critical box lying between the first and third column, this box can be broken into two critical boxes, one lying between the first two columns and the other lying between the last two columns. Lemma 3.15 shows that when there is a critical box with a feasible machine index i_s left between the starting and ending nodes, this critical box can be broken into two critical boxes, one starting with the feasible index i_s and the other ending with the feasible index i_s.

Definition 3.4. *In the network representation of the m machine* 3 *sublot problem, two subpaths* $\{(i_s, 1) - (i_s, 2) - (i_s, 3) - (i_{s+1}, 3) - \ldots - (i_t, 3)\}$ *and* $\{(i_s, 1) - (i_s+1, 1) - (i_s+2, 1) - \ldots - \ldots - (i_t, 1) - (i_t, 2) - (i_t, 3)\}$ *are said to form a critical box if each subpath is part of a critical path. Any box in the representation of the m machine* 3 *sublot problem is identified by its upper-left and lower-right corner nodes* ($(i_s, 1)$ *and* $(i_t, 3)$ *in the box defined above.*).

Lemma 3.14. *In an optimal solution for the m machine* 3 *sublot problem if nodes* $(i_s, 1)$ *and* $(i_t,3)$ *define a critical box, then the boxes* $\{(i_s, 1) - (i_t, 2)\}$and $\{(i_s, 2) - (i_t, 3)\}$ *are also critical boxes.*

Lemma 3.15. *For a given vector of sublot sizes,* $\mathbf{x}^*$, *if the node pairs* $\{(i_r, 1) - (i_t, 2)\}$ (or $\{(i_r, 2) - (i_t, 3)\}$ *form a critical box between the critical indices* i_r *and* i_t *and if* i_s *is also a critical index with* $i_r < i_s < i_t$, *then the boxes formed by* $\{(i_r, 1) - (i_s, 2)\}$ *and* $\{(i_s, 1) - (i_t, 2)\}$ (or $\{(i_r, 2) - (i_s, 3)\}$ *and* $\{(i_s, 2) - (i_t, 3)\}$) *are also critical.*

The following lemma, its corollary, and theorem guarantee that the critical boxes defined above will always exist in the network for the optimal set of sublot sizes.

Lemma 3.16. *There exists an optimal solution for the m machine* 3 *sublot lot streaming problem which always provides at least one critical box between each pair of consecutive columns.*

Corollary 3.2. *There exists an optimal solution for the m machine* 3 *sublot lot streaming problem where critical boxes are formed between feasible machine indices that point to each other.*

Theorem 3.8. *There exists an optimal solution to the m machine* 3 *sublot lot streaming problem such that either there exists a pair of adjacent feasible indices* i_r *and* i_s *forming critical boxes between columns* 1 *and* 2, *and between columns* 2 *and* 3, *or there exists two pairs of adjacent feasible indices* (i_r, i_s) *and* (i_t, i_q), *for* $i_r \rightarrow i_s$, $i_t \rightarrow i_q$, $i_r \leq i_s \leq i_t \leq i_q$, *such that a critical box* $\{(i_r, 1), (i_s, 2)\}$ *is formed between columns* 1 *and* 2, *and a critical box* $\{(i_t, 2), (i_q, 3)\}$ *is formed between column* 2 *and* 3.

After generating the optimal pattern of the critical paths in the network, Williams et al. [Wil97] provided the information in Theorem 3.8 that obtaining a solution in terms of the ratios of sublot sizes yields a unique solution in terms of the sublot sizes, and vice versa. In order to do that, they showed there is a one-to-one correspondence between the domains X and Z.

Theorem 3.9. *There is a one-to-one correspondence between the domains X and Z where*

$$X = \{(x_1, x_2, x_3)^T | x_1 + x_2 + x_3 = 1 \text{ and } 0 < x_i < 1, \forall i\}$$

and

$$Z = \{(z_1, z_2)^T | \ 0 < z_1, \ 0 < z_2\}$$

where the transformation from X to Z is $z_1 = x_1/x_2$ *and* $z_2 = x_2/x_3$.

After showing that the *m* machine 3 sublot problem is also solvable in *Z* domain, Williams et al. [Wil97] defined the length of paths in the network in terms of z_1 and z_2. First they generated the formulas to obtain the sublot sizes when z_1 and z_2 are known, such as,

$$x_1 = \frac{z_1 z_2}{z_1 z_2 + z_2 + 1}, \tag{3.54}$$

$$x_2 = \frac{z_2}{z_1 z_2 + z_2 + 1}, \tag{3.55}$$

and

$$x_3 = \frac{1}{z_1 z_2 + z_2 + 1}. \tag{3.56}$$

Then, they let $x_3 = 1 - x_1 - x_2$ and inserted the new values for x_1, x_2 in (3.51) and to obtain the following,

$$M_{ij}(z_1, z_2) = \frac{a_{ij} z_1 z_2 + b_{ij} z_2}{z_1 z_2 + z_2 + 1} + T(j, m), \tag{3.57}$$

where $a_{ij} = T(1, i) - T(j, m)$ and $b_{ij} = T(i, j) - T(j, m)$. Therefore, the longest path for any z_1 and z_2 will be

$$M(z_1, z_2) = \max_{i \le j} \{M_{ij}(z_1, z_2)\}, \tag{3.58}$$

which is defined over the domain $Z = \{\mathbf{z} \in R^2 |\ 0 < z_1,\ 0 < z_2\}$. It is observed that the function described by (3.57) is the upper envelope of the intersecting hyperplanes $M_{ij}(z_1, z_2)$, $i \le j$. Finally, the optimal makespan, M^*, is obtained from the following,

$$M^* = \min_{z \in Z} \{M(z_1, z_2)\}\ . \tag{3.59}$$

Williams et al. [Wil97] stated that some important characteristics of the function $M_{ij}(z_1, z_2)$ must be known in order to obtain the optimal solution. They showed that the function is monotone in both z_1 and z_2 directions, as stated in Theorem 3.10 below.

Theorem 3.10. *The function $M_{ij}(z_1, z_2)$ is a monotone function in z_1 and z_2.*

In fact, the partial derivatives of the function $M_{ij}(z_1, z_2)$, reveal the property mentioned in the above theorem, where

$$\frac{\partial M_{ij}(z_1, z_2)}{\partial z_1} = \frac{a_{ij} z_2 (z_2 + 1) - b_{ij} z_2^2}{(z_1 z_2 + z_2 + 1)^2} \tag{3.60}$$

is invariant to z_1 and

$$\frac{\partial M_{ij}(z_1, z_2)}{\partial z_2} = \frac{a_{ij} z_1 + b_{ij}}{(z_1 z_2 + z_2 + 1)^2} \tag{3.61}$$

is invariant to z_2. Following to the above theorem, Williams et al. [Wil97] determined that the optimal solution of $M_{ij}(z_1, z_2)$ occurs at the lower and upper bounds of z_1 and z_2.

Lemma 3.17. *The minimum value of $M_{ij}(z_1, z_2)$ occurs at the lower and upper bounds of z_1 and z_2.*

Since z_1 and z_2 are assigned either their lower or upper limits which are defined through some (i, j), $i \le j$, indices, a particular value of z_1 and z_2 can also be represented by $z_1(i, j)$ and $z_2(i, j)$, respectively. The above lemma shows that the minimum makespan can

be obtained by a complete enumeration. However, for any feasible index pair (i, j) four combinations of $z_1(i, j)$ and $z_2(i, j)$ need to be considered. Since the network representation of the m machine 3 sublot lot streaming problem has $m\ (m + 1) / 2$ paths, therefore, there are potentially $O(m^2)$ pairs of feasible indices. Hence, the complete enumeration is essentially an exhaustive search with $O(m^2)$ time complexity.

Williams et al. [Wil97] showed that a better algorithm could be designed by using the properties stated in Theorem 3.10 and Lemma 3.17. Since the first and the last machine indices are always feasible, they started their algorithm with initial feasible indices $i_1 = 1$ and $i_r = m$, that is, they used the initial critical boxes $\{(i_1, 1) - (i_2, 2)\}$ and $\{(i_{r-1}, 1) - (i_r, 2)\}$ to evaluate

$$z_1 = \frac{T(i_1, i_2 - 1)}{T(i_1 + 1, i_2)}$$

and

$$z_2 = \frac{T(i_{r-1}, i_r - 1)}{T(i_{r-1} + 1, i_r)}.$$

With this starting point and due to the monotonicity of the function, Williams et al. stated that their algorithm behaves as a coordinated descent procedure to locate the optimal critical boxes between columns 1 and 2, and between columns 2 and 3 in at most $r \leq m$ iterations, where r is the number of feasible indices in the problem. Finally, their algorithm is as follows,

m x 3 Search Algorithm

1. Perform feasible index search, to obtain r feasible indices in the m machine 3 sublot lot streaming problem. Denote these indices $i_1\ (= 1),\ i_2,\ \cdots,\ i_r\ (= m)$, where

$i_1 < i_2 < \cdots < i_r$, and where i_k and i_{k+1} are adjacent feasible indices for k = 1, 2, ..., r-1.

Set s = 1 and $t = r$-1. Let

$$z_1 = \frac{T(i_s, i_{s+1} - 1)}{T(i_s + 1, i_{s+1})}$$

and

$$z_2 = \frac{T(i_t, i_{t+1} - 1)}{T(i_t + 1, i_{t+1})},$$

go to Step 3.

2a. Set

$$z_1 = \frac{T(i_s, i_{s+1} - 1)}{T(i_s + 1, i_{s+1})}$$

and go to Step 4.

2b. Set

$$z_2 = \frac{T(i_t, i_{t+1} - 1)}{T(i_t + 1, i_{t+1})}.$$

3. If $\partial M_{i_s i_{s+1}}(z_1, z_2)/\partial z_1 < 0$ and $\partial M_{i_{s+1} i_{t+1}}(z_1, z_2)/\partial z_1 < 0$, set $s = s + 1$.

   Go to Step 2a.

4. If $\partial M_{i_s i_t}(z_1, z_2)/\partial z_2 > 0$ and $\partial M_{i_s i_{t+1}}(z_1, z_2)/\partial s z_2 > 0$, set $t = t$ - 1 and go to Step 2b.

   Else, if $\partial M_{i_s i_{t+1}}(z_1, z_2)/\partial z_1 < 0$ and $\partial M_{i_{s+1} i_{t+1}}(z_1, z_2)/\partial z_1 > 0$, the solution is optimal, go to Step 5.

5. The optimal sublot sizes are

$$x_1^* = \frac{z_1 z_2}{z_1 z_2 + z_2 + 1},$$

$$x_2^* = \frac{z_2}{z_1 z_2 + z_2 + 1},$$

$$x_3^* = \frac{1}{z_1 z_2 + z_2 + 1},$$

and the optimal makespan is $M^* = T(1,i_s)x_1 + T(i_s,i_t)x_2 + T(i_t,m)x_3$.

Williams et al. [Wil97] also stated that the complexity of the above algorithm is $O(m)$, when the feasible indices are known. However, if the feasible indices also need to be determined, the overall complexity becomes $O(m^2)$ due to the Feasible Search Algorithm.

3.6 Solution of the $F, m \mid S, II \mid n, C, R \mid C_{max}$ Problem

3.6.1 Williams, Tüfekçi and Akansel's Heuristic

Williams, Tüfekçi and Akansel [Wil97] developed an efficient heuristic for the general size lot streaming problem, which will be also referred to as "m x n lot streaming problem", using the optimal solution they obtained for the same problem data with three sublots. This method provided the closest approximation to the optimal solution at the time there were no efficient ways of obtaining the optimal solution other than solving the LP model.

Williams et al. [Wil97] first generated the formulas to obtain the sizes of the n sublots when the ratios of the sizes of the consecutive sublots are known. As seen before,

in the two sublot problem the sublot sizes are obtained from the sublot size ratio as follows,

$$x_1 = \frac{z}{z+1}$$

and

$$x_2 = \frac{1}{z+1},$$

where $z = x_1 / x_2$. Meanwhile, in the three sublot problem the sublot sizes have been calculated through the following formulas

$$x_1 = \frac{z_1 z_2}{z_1 z_2 + z_2 + 1},$$

$$x_2 = \frac{z_2}{z_1 z_2 + z_2 + 1},$$

and

$$x_3 = \frac{1}{z_1 z_2 + z_2 + 1},$$

when the ratios $z_1 = x_1 / x_2$ and $z_2 = x_1 / x_2$ have already been obtained. Finally, the above formulas can be generalized to the m machine n sublot problem as follows,

$$x_1 = \frac{\prod_{j=1}^{n-1} z_j}{\sum_{k=1}^{n-1} \prod_{j=k}^{n-1} z_j + 1}, \tag{3.62}$$

$$x_i = \frac{\prod_{j=i}^{n-1} z_j}{\sum_{k=1}^{n-1} \prod_{j=k}^{n-1} z_j + 1}, \qquad 2 \le i \le n-1, \tag{3.63}$$

and

$$x_n = \frac{1}{\sum_{k=1}^{n-1}\prod_{j=k}^{n-1} z_j + 1}, \tag{3.64}$$

where $z_k = x_k / x_{k+1}$, $k = 1,\ldots, n\text{-}1$.

Williams et al. [Wil97] stated that it was expected that the m machine n sublot was also solvable through the critical boxes between the consecutive column of the network and the corresponding z_k values. They also observed that in the optimal solution of the m machine n sublot problem there existed some blocks of columns using the same sublot size ratio and the number of such blocks rarely exceeded 2. They also observed that when there were two blocks of columns, the first block and the second block used the z_1^* and z_2^* values, respectively, which are the ratios of the sizes of the consecutive sublots in the optimal solution of the m machine 3 sublot lot streaming problem.

Based on their observations mentioned above, Williams et al. [Wil97] developed a heuristic procedure for the m machine n sublot lot streaming problem. Their algorithm receives the values m, n and unit processing times p_i, $i = 1, \ldots, m$ as inputs, then calls the optimal m x 3 Search Algorithm as a subroutine to obtain $z_1 = x_1 / x_2$ and $z_2 = x_2 / x_3$. Following this step, the algorithm continues with locating the best place to form the two blocks in the network. William's et al.'s two block heuristic algorithm for the m machine n sublot problem is as follows,

Two-Block Heuristic Algorithm for the m x n Problem

0. Read in m, n, p_i, $i = 1, 2, \ldots, m$.
1. Call m x 3 Search Algorithm to obtain the optimal values of x_1, x_2 and x_3.

   Let

$$zz_1 = x_1 / x_2$$
$$zz_2 = x_2 / x_3$$

Set MINSPAN = ∞, $k = 1$ and $kk = 0$.

2. If $k = n - 1$ go to Step 5.

   Let

   $$z_1 = z_2 = \ldots = z_k = zz_1$$

   and

   $$z_{k+1} = z_{k+2} = \ldots = z_{n-1} = zz_2.$$

3. For $i = 1$ to n do

   Compute x_i by using Equation 3.62.

4. By using x_i values compute the makespan $M_k(\mathbf{x})$.

   If $M_k(\mathbf{x})$ < MINSPAN then

   $$\text{MINSPAN} = M_k(\mathbf{x})$$

   $$kk = k$$

   Endif.

   $$k = k + 1$$

   Go to Step 2.

5. Print values of MINSPAN, kk, x_i, $i = 1, 2, \ldots, n$.

   End.

The main body of the above heuristic iterates $O(n)$ times to check every sublot if it might be break-point between the blocks. However, the calculation of the $M_k(\mathbf{x})$ in Step 4 can be accomplished in $O(mn)$ with a longest path algorithm. Therefore, Williams et al. [Wil97] reported that the overall complexity of their heuristic algorithm was $O(mn^2)$.

The above heuristic algorithm was also tested on 155 randomly generated sample lot streaming problems. The algorithm found optimal solutions to 36 problems. The heuristic also delivered quite satisfactory results in terms of the error-bounds, where the worst-case error and the average error had been only 5.68 percent and 0.76 percent, respectively [Wil97].

3.6.2 An $O(n^3 * (m + n))$ Algorithm for the m x n Problem

The network representation developed by Baker and Pyke [Bak90] for single-job lot streaming problems in flow shops constitutes the core of the algorithm that will be presented in this section. The linear programming formulation that led to the development of the network representation was provided at the beginning of this chapter. The same project network representation will be replicated in Figure 3.5 below.

The makespan in the original lot streaming problem corresponds to the length of the longest path in the network, however, since the sublot sizes are decision variables of both problems, the network also has variable-length activity times. Therefore, the problem of minimizing the makespan involves allocating work to sublots to minimize the length of the longest path in the network.

A characterization of the paths in the network representation is necessary to initiate the development of the techniques to obtain the minimal longest path. This characterization is provided in Definition 3.5.

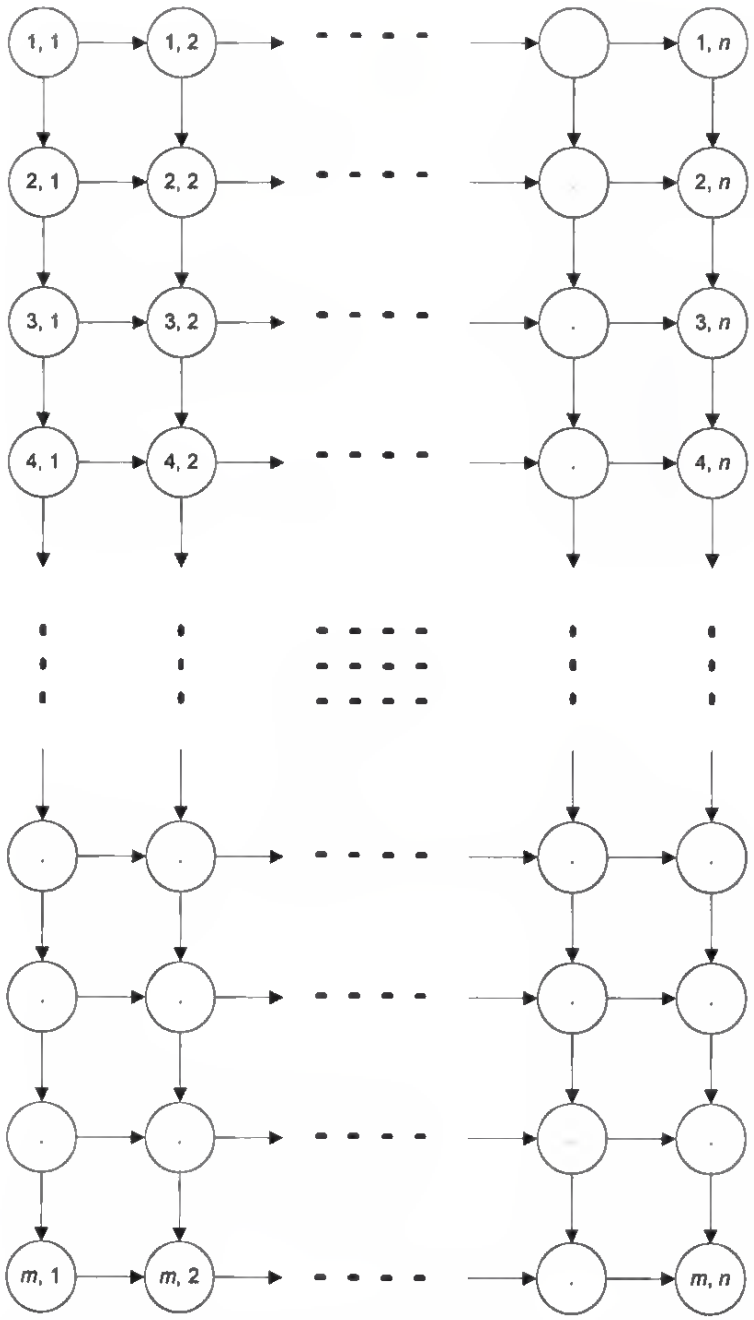

Figure 3.5 Network representation of the m x n lot streaming problem

Definition 3.5. *Let* $I[j]$, $j = 1, 2, \ldots, n\text{-}1$, *denote a machine index in the m x n lot streaming problem. The set of machine indices I,*

$$I = \{I[1], I[2], \ldots, I[n\text{-}1]\}$$

where $I[j] \le I[j+1]$, *for* $j = 1, \ldots, n\text{-}2$, *defines the path, denoted by* $P(I)$, *that crosses over from column j to column* $j+1$ *in the network representation of the problem at node* $(I[j], j)$, *for* $j = 1, \ldots, n\text{-}1$. *This path,* $P(I)$, *also traverses the sequence of nodes* $(I[j-1], j)$ *through* $(I[j], j)$, *for* $j = 1, \ldots, n$, *where* $I[0] \equiv 1$ *and* $I[n] \equiv m$.

Following Definition 3.5, the node labeled $(I[j], j)$ on the path $P(I)$ is called a "crossover" node since the path crosses over from column j to column $j+1$ at this node of the network. For a given vector of sublot sizes, the length of the path $P(I)$, denoted by $L_{P(I)}(\mathbf{x})$, is expressed as

$$L_{P(I)}(\mathbf{x}) = \sum_{j=1}^{n} T(I[j-1], I[j])x_j ,$$

where $T(i, j) = \sum_{k=i}^{j} p_k$ and p_k is the unit processing time on machine k. If this path happens to be the longest path in the network for a given vector of sublot sizes, its length is the makespan of the lot streaming problem for the given vector of sublot sizes. In general, the makespan of the lot streaming problem is the following,

$$M_x(\mathbf{x}) = \max_{I \in K} L_{P(I)}(\mathbf{x})$$

where the set K contains every set of n-1 machine indices $\{I[1], I[2], \ldots, I[n\text{-}1]\}$ such that $I[j] \le I[j+1]$, for $j = 1, 2, \ldots, n\text{-}2$. This expression provides the information that the makespan function is the upper envelope of the intersection of the hyperplanes $L_{P(I)}(\mathbf{x})$,

$I \in K$, and therefore $M_x(\mathbf{x})$ is a convex function. Meanwhile, the makespan function, $M_x(\mathbf{x})$, is defined in the convex domain

$$X_D = \left\{ \mathbf{x} \in R^n | x_1 + x_2 + \ldots + x_n = 1,\ 0 \le x_j \le 1,\ j = 1, \ldots, n \right\},$$

since sum of the sublot sizes must be equal to the size of the whole production lot (which is 1 because of normalization). Since the goal is to obtain the longest path with the minimal length, the optimal makespan is expressed by the following equation,

$$M^* = \min_{x \in X_D} M_x(\mathbf{x})$$

or

$$M^* = \min_{x \in X_D} \max_{I \in K} L_{P(I)}(\mathbf{x}).$$

The final expression reveals that we need to minimize a convex function over a convex domain. In this case, when a local minima is located, it is guaranteed to be the global minima.

Theorem 3.11 below specifies the conditions under which the path defined by any set of n-1 machine indices I is the longest path in the network.

Theorem 3.11. *Each index* $I[j]$ *in the set* $I = \{I[1], I[2], \ldots, I[n-1]\}$ *where* $I[j] \le I[j+1]$ *for* $j = 1, \ldots, n$-2, *that defines a longest path must satisfy the condition*

$$\frac{T(i, I[j]-1)}{T(i+1, I[j])} \le z_j \le \frac{T(I[j], i'-1)}{T(I[j]+1, i')} \tag{3.65}$$

where $I[j\text{-}1] \le i \le I[j]$ -1 *with* $I[j] \ne 1$ *and* $I[j] + 1 \le i' \le I[j+1]$ *with* $I[j] \ne m$, *and where* $z_j = x_j / x_{j+1}$ *for* $j = 1, \ldots, n-1$.

Proof: Let $I = \{I[1], I[2], \ldots, I[n\text{-}1]\}$ be the set of machine indices defining a longest path in the network. In order that $P(I)$ might be the longest path in the network, the following inequality must be satisfied,

$$\sum_{k=1}^{n} T(I[k-1], I[k])x_k \geq \sum_{k=1}^{j-1} T(I[k-1], I[k])x_k + T(I[j-1], i)x_j + T(i, I[j]-1)x_{j+1} \quad (3.66)$$

$$+ \sum_{k=j+1}^{n} T(I[k-1], I[k])x_k$$

for $j = 1,\ldots, n\text{-}1$, $I[j\text{-}1] \leq i \leq I[j]\text{-}1$ and $I[j] \neq 1$. Equation 3.66 reduces to the inequality below,

$$T(I[j-1], I[j])x_j \geq T(I[j-1], i)x_j + T(i, I[j]-1)x_{j+1},$$

which is further simplified to

$$T(i+1, I[j])x_j \geq T(i, I[j]-1)x_{j+1}.$$

It finally leads to the following lower bound on the ratio of sublot sizes x_j and x_{j+1},

$$\frac{x_j}{x_{j+1}} \geq \frac{T(i, I[j]-1)}{T(i+1, I[j])} \quad (3.67)$$

for $j = 1, \ldots, n\text{-}1$, $I[j\text{-}1] \leq i \leq I[j]\text{ -}1$ and $I[j] \neq 1$.

In order that $P(I)$ might be the longest path in the network, the following inequality must also be satisfied,

$$\sum_{k=1}^{n} T(I[k-1], I[k])x_k \geq \sum_{k=1}^{j} T(I[k-1], I[k])x_k + T(I[j]+1, i)x_j + T(i, I[j+1])x_{j+1} \quad (3.68)$$

$$+ \sum_{k=j+2}^{n} T(I[k-1], I[k])x_k$$

for $j = 1,\ldots, n\text{-}1$, $I[j] + 1 \leq i \leq I[j\text{+}1]$ and $I[j] \neq m$. Equation 3.68 reduces to the inequality below,

$$T(I[j], I[j+1])x_{j+1} \geq T(I[j]+1, i)x_j + T(i, I[j+1])x_{j+1},$$

which is further simplified to

$$T(I[j],i-1)x_{j+1} \geq T(I[j]+1,i)x_j .$$

It finally leads to the following upper bound on the ratio of sublot sizes x_j and x_{j+1},

$$\frac{x_j}{x_{j+1}} \leq \frac{T(I[j],i-1)}{T(I[j]+1,i)} \tag{3.69}$$

for j =1, …, n-1, $I[j] + 1 \leq i \leq I[j+1]$ and $I[j] \neq m$.

Using the definitions $z_j \equiv x_j / x_{j+1}$ *for* $j = 1,\ldots,n-1$, Equations 3.67 and 3.69 can be combined in the following inequality,

$$\frac{T(i,I[j]-1)}{T(i+1,I[j])} \leq z_j \leq \frac{T(I[j],i'-1)}{T(I[j]+1,i')} \tag{3.70}$$

for j = 1, …, n-1, $I[j\text{-}1] \leq i \leq I[j]$ -1, $I[j] \neq 1$ and $I[j] + 1 \leq i' \leq I[j+1]$, $I[j] \neq m$. Hence, Equation 3.65 must be satisfied for every index of the set I, in order for the path $P(I)$ to define a longest path in the network. □

As stated by Theorem 3.11, every index in the set I must satisfy Equation 3.65 if the path $P(I)$ is a longest path in the network. Since the overall goal is to obtain the longest path with the minimal length, a method is needed to determine which machine indices are eligible to define a longest path in the network. Lemma 3.18 below shows that machine indices 1 and m are always in the set of machine indices defining a longest path.

Lemma 3.18. *Machine index* 1 *or* m *can always be used to define longest paths in the network representation of the* m x n *lot streaming problem.*

Proof: Simply let x_1 = 1 - ε, and x_2 = … = x_n = ε /(n-1). It is easy to see that the longest path travels nodes (1, 1) - (2, 1) - ... - (m, 1) - (m, 2) - … - (m, n), for $\varepsilon > 0$ and small enough. Similarly, for x_j = 1 - ε, and $x_1 = x_2 = \ldots = x_{j-1} = x_{j+1} = \ldots = x_n = \varepsilon/(n\text{-}1)$, $1 \leq j \leq n$, the longest path travels nodes (1, 1) - (1, 2) - … - (1, j) - (2, j) - ... - (m, j) -

$(m, j+1)$ - … - (m, n), for $\varepsilon > 0$ and small enough. Finally, for $x_n = 1 - \varepsilon$, and $x_1 = x_2 = \ldots = x_{n-1} = \varepsilon/(n-1)$, the longest path travels nodes (1, 1) - (1, 2) - (1, n) - (2, n) - ... - (m, n), for $\varepsilon > 0$ and small enough. □

At this point, Equation 3.65 only provides the necessary conditions for the machine index set I to define a longest path in the network representation of the m x n lot streaming problem since they are derived by comparing only a subset of all possible paths in the network to the path defined by the machine index set I. However, Lemmas 3.19-3.23 and Theorem 3.12 provide necessary and sufficient conditions for the machine index set I to define a longest path in the network representation of the m x n lot streaming problem.

The concept called "feasible indices" will also be particularly helpful in this section. As previously described, feasible indices are the machine indices with a possibility of defining a longest path in the network. The short algorithm that obtains the set of feasible indices is given in Section 3.5, where the m machine 3 sublot problem is discussed. The definitions and lemmas pertaining to feasible indices that were provided earlier are also valid in this section.

In the following five lemmas, we will characterize the longest paths in an optimal solution to the m x n problem. In the proofs of these lemmas, we assume that the feasible index set associated with the given instance of an m x n problem is already at hand. Let $F = \{1 = i_1, i_2, \ldots, i_k = m\}$ represent this set.

Lemma 3.19. *For any given feasible vector* $\mathbf{x} = (x_1, x_2, \ldots, x_n)^T$, *the indices of any longest path are nondecreasing both in machine index and in sublot index.*

Proof: From the network representation of the m x n problem, the arcs in the m x n network are either in the form $[(i,j) \rightarrow (i+1,j)]$ or in the form $[(i,j) \rightarrow (i,j+1)]$. □

Lemma 3.20. *If* x* *is an optimal solution to an m* x *n problem, then there exists at least two distinct longest paths in the network representation of the problem.*

Proof: Assume that the longest path is unique with crossover indices set I, for $1 \le I[1] \le \ldots \le I[n-1] \le m$. Consider the quantities T(1, I[1]), $T(I[1], I[2])$, … , and $T(I[n-1], m)$. Without loss of generality assume that $T(1, I[1]) \ge T(I[1], I[2])$. Consider the following modifications $x_2 = x_2^* + \delta$, $x_1 = x_1^* - \delta$, and $x_3 = x_3^*, \ldots, x_n = x_n^*$ for $\delta > 0$ and small enough. Note that $x_1 + x_2 + x_3 + \ldots + x_n = 1$ and the difference between the length of the critical path for x* and for **x** is given by

$$M^* - M = \delta\ (T(1, I[1]) - T(I[1], I[2])) \ge 0$$

for $I[1] > 1$. Here, we assume $T(k, t) \equiv 0$ if $k > t$. Therefore, if the inequality is strict, then for $\delta > 0$ and small enough, we can reduce the makespan, a contradiction to the optimality of **x***. In the case of equality, we can increase δ until another path becomes critical. If $T(1, I[1]) \le T(I[1], I[2])$, we can simply add δ to x_2^* and subtract δ from x_1^*, the conclusion will follow similarly. For $I[1] = \ldots = I[n-1] = 1$, we can set $x_{n-1} = x_{n-1}^* + \delta$, $x_n = x_n^* - \delta$ and $x_1 = x_1^*, \ldots, x_{n-2} = x_{n-2}^*$. For $I[1] = \ldots = I[n\text{-}1] = m$, we can set $x_1 = x_1^* - \delta$, $x_2 = x_2^* + \delta$ and $x_3 = x_3^*, \ldots, x_n = x_n^*$. For $\delta > 0$ and small enough, the conclusion will again follow. □

Definition 3.6. *In the network representation of the m* x *n problem, two subpaths* $\{(I[i], i) - (I[i], i+1) - \ldots (I[i], j) - (I[i]+1, j) - \ldots - (I[j], j)\}$ *and* $\{(I[i], i) - (I[i]+1, i) - \ldots - (I[j], i) - (I[j], i+1) - \ldots - (I[j], j)\}$, *for* $1 \le i < j \le n$ *and* $1 \le I[i] < I[j] \le m$, *are said to*

form a critical box if each subpath is a part of a critical path. We identify any box in the network representation of the m x n problem by its upper-left and lower-right corner nodes ((I[i], i) and (I[j], j) in the box defined above).

The next three lemmas define some significant properties of critical boxes.

Lemma 3.21. *In an optimal solution for the m x n problem if {(I[i], i) - (I[j], j)} form a critical box, then the boxes {(I[i], i) - (I[j], i+1)}, {(I[i], i+1) - (I[j], i+2)}, ... , and* $\{I[i], j-1)-(I[j], j)\}$, *for* $1 \le i < j \le n$ *and* $1 \le I[i] < I[j] \le m$, *are also critical boxes.*

Proof: Assume that the box {(I[i], i) - (I[j], i+1)} is not critical. From the statement we have

$$T(I[i]+1, I[j])x_i + p_{I[j]}x_{i+1}+\ldots+p_{I[j]}x_{j-1} = p_{I[i]}x_{i+1} + p_{I[i]}x_{i+2}+\ldots+T(I[i], I[j]-1)x_j$$

Let $\mathbf{x}^*$ be the optimal solution that yields the critical box. Since the box {(I[i], i) - $(I[j],i+1)$} is not critical, the subpath {(I[i], i) - (I[i], i+1) - ... - (I[j], i+1)} is not critical. Consider the following changes.

$x_i = x_i^* - \delta/2$, $x_{i+1} = x_{i+1}^* + \delta$, $x_j = x_j^* - \delta/2$, and $x_k = x_k^*$, for $k \neq i$, $k \neq i+1$, $k \neq j$.

Note that $x_1 + x_2 + \ldots + x_n = 1$. Furthermore, the length of the critical subpaths forming the critical box are strictly reduced for $\delta > 0$ and small enough while the length of the noncritical path {(I[i], i) - (I[i], i+1) - ... - (I[j], i+1)} increases. For $\delta > 0$ and small enough, the new solution reduces the makespan, a contradiction. The same justification applies for boxes {(I[i], i+1) - (I[j], i+2)},..., and {(I[i], j-1) - (I[j], j)} as well.□

The following lemma provides the information that critical boxes are formed between two consecutive feasible machine indices.

Lemma 3.22. *For a given vector of optimal sublot sizes,* $\mathbf{x}^*$, *if the node pairs* $\{(I[i],i) - (I[i+1], i+1)\}$ *form a critical box between the critical indices* $I[i]$ *and* $I[i+1]$ *and if* $I[k]$ *is also a critical index with* $I[i] < I[k] < I[i+1]$, *then the boxes formed by* $\{(I[i], i) - (I[k], i+1)\}$ *and* $\{(I[k], i) - (I[i+1], i+1)\}$ *are also critical.*

Proof: Assume that $I[k]$ is not a crossover index. From the criticality of the original box $\{(I[i], i) - (I[j], i+1)\}$, we have

$$\frac{T(I[i], I[i+1]-1)}{T(I[i]+1, I[i+1])} = \frac{x_i}{x_{i+1}} = z_i \tag{3.71}$$

or,

$$T(I[i]+1, I[k])x_i + T(I[k]+1, I[i+1])x_i = T(I[i], I[k]-1)x_{i+1} + T(I[k], I[i+1]-1)x_{i+1}. \tag{3.72}$$

From the assumption, we also have

$$T(I[i]+1, I[k])x_i < T(I[i], I[k]-1)x_{i+1} \tag{3.73}$$

or,

$$z_i = \frac{x_i}{x_{i+1}} < \frac{T(I[i], I[k]-1)}{T(I[i]+1, I[k])}. \tag{3.74}$$

Furthermore, from the construction of feasible indices we have

$$\frac{T(I[i], I[k]-1)}{T(I[i]+1, I[k])} = \min_{I[i]<w\le m} \frac{T(I[i], w-1)}{T(I[i]+1, w)}. \tag{3.75}$$

For $w = I[k]$ and using Equation 3.71 we can write Equation 3.75 as

$$\frac{T(I[i], I[k]-1)}{T(I[i]+1, I[k])} = \min_{I[i]<w\le m} \frac{T(I[i], w-1)}{T(I[i]+1, w)} \le \frac{T(I[i], I[i+1]-1)}{T(I[i]+1, I[i+1])} = z_i. \tag{3.76}$$

Combining Equations 3.71, 3.74 and 3.76 we get $z_i < z_i$,

a contradiction.

Note that Equations 3.74 and 3.76 can only be satisfied as exact equality, if and only if, $I[k]$ is also a crossover index. □

The following lemma ensures that, in the optimal solution to an m x n lot streaming problem, there exists at least one critical box between each consecutive columns.

Lemma 3.23. *There exists an optimal solution for the m x n lot streaming problem which always provides at least one critical box between each pair of consecutive columns.*

Proof: From Lemma 3.21, we know that if there is a critical box between columns i and j, $1 \le i < j \le n$, there are also critical boxes between every two consecutive columns from column i to column j. Therefore, the only situation not covered by the previous lemmas is the case where either there exists a box between columns 1 and j, $1 \le j < n$ and none between columns j and n or there exists a box between columns j and n, $1 \le j < n$, and none between columns 1 and j. We will prove the later case here, since the proof of the former case is identical.

Let $\mathbf{x}^*$ be the optimal solution and assume there exists a critical box only between columns j and n, between the crossover indices i_s and i_t, $i_s < i_t$. Consider the following; modification to the $\mathbf{x}^*$.

$$x_i = x_i^* - \frac{r}{j-1}\sum_{k=j}^{n} x_k^*, \ \forall\, i = 1,\ldots,j-1,$$

and

$$x_i = (r+1)x_i^*, \ \forall\, i = j,\ldots,n.$$

Note that $x_1 + x_2 + \ldots + x_n = 1$. After this modification, any subpath from a node (i_s, j) to a node (i_t, n), $1 \le i_s < i_t \le m$ is scaled by $(1+r)$.

Therefore, the critical subpaths between columns j and n maintain their status as critical for all values of $-1 < r < \min_{i=1,\ldots,j-1}\left\{(j-1)x_i^* \Big/ \sum_{k=j}^{n} x_k^*\right\}$. Further note that, with this scaling, the length of the critical subpath from node (1,1) to node (i_s, j) is modified by $-\frac{r}{j-1}\sum_{k=j}^{n} x_k^* L_{\{(1,1),(i_s,j)\}}$, where $L_{\{(1,1),(i_s,j)\}}$ is the length of the original subpath from (1, 1) to (i_s, j). Thus, the net change in the lengths of both critical paths is given by

$$L' - L^* = \delta L = -\frac{r}{j-1}\sum_{k=j}^{n} x_k^* L_{\{(1,1),(i_s,j)\}} + rL_{\{(i_s,j),(m,n)\}}.$$

Therefore, for

$$\frac{1}{j-1}\sum_{k=j}^{n} x_k^* L_{\{(1,1),(i_s,j)\}} < L_{\{(i_s,j),(m,n)\}}$$

by selecting -1 < r < 0 and large enough, we reduce the length of the critical paths, a contradiction. Similarly, for

$$\frac{1}{j-1}\sum_{k=j}^{n} x_k^* L_{\{(1,1),(i_s,j)\}} > L_{\{(i_s,j),(m,n)\}}$$

by selecting $\min_{i=1,\ldots,j-1}\left\{x_i^* \Big/ \sum_{k=j}^{n} x_k^*\right\} > r > 0$ and small enough, we once again can reduce the length of the critical paths, a contradiction. If

$$\frac{1}{j-1}\sum_{k=2}^{n} x_k^* L_{\{(1,1),(i_s,j)\}} = L_{\{(i_s,j),(m,n)\}}$$

then, as we increase $r > 0$, eventually, for a particular value of r, say r', a new subpath between columns 1 and j will become critical at a crossover index i_q, while makespan

remaining the same. Together with i_s these two indices will form a critical box between columns 1 and j. □

An immediate consequence of these lemmas is,

Corollary 3.3. *There exists an optimal solution for the m x n lot streaming problem where critical boxes are formed between feasible machine indices that point to each other.*

Finally, the following theorem completely characterizes the optimal solution in an m x n lot streaming problem.

Theorem 3.12. *There exists an optimal solution to the m x n lot streaming problem such that there are pairs of adjacent feasible indices forming boxes between adjacent columns.*

Proof: Consequences of Lemma 3.19-23, and Corollary 3.3. □

Because $x_1 = z_1 x_2$, $x_2 = z_2 x_3$, …, $x_{n-1} = z_{n-1} x_n$ and $x_1 + x_2 + \ldots + x_n = 1$, solving for the sublot sizes in terms of z_i, $1 \le i \le n\text{-}1$, results in the following expressions for x_i, $1 \le i \le n$,

$$x_1 = \frac{\prod_{j=1}^{n-1} z_j}{\sum_{k=1}^{n-1} \prod_{j=k}^{n-1} z_j + 1} \tag{3.77}$$

$$\vdots$$

$$x_i = \frac{\prod_{j=i}^{n-1} z_j}{\sum_{k=1}^{n-1} \prod_{j=k}^{n-1} z_j + 1}, \qquad 2 \le i \le n-1, \tag{3.78}$$

$$\vdots$$

$$x_n = \frac{1}{\sum_{k=1}^{n-1}\prod_{j=k}^{n-1} z_j + 1} \tag{3.79}$$

Then, the length of the longest path can be expressed in terms of z_i, $1 \le i \le n$-1. At this point, a method of obtaining the correct set of sublot sizes that correspond to these ratios is needed if more than one set of sublot sizes exist for each set of ratios. However, by Theorem 3.13 given below, there is a one to one correspondence between the domains

$$X = \left\{(x_1, x_2, \ldots, x_n)^T \mid x_1 + x_2 + \ldots + x_n = 1 \text{ and } 0 < x_i < 1, \forall i\right\}$$

and

$$Z = \left\{(z_1, z_2, \ldots, z_{n-1})^T \mid 0 < z_i, \forall i\right\}.$$

As a result, once z_i's, $1 \le i \le n$-1, are specified, unique values of the sublot sizes, x_i, $1 \le i \le n$, can be determined. The converse is also true. Because of this, the m x n lot streaming problem can be discussed either in terms of the sublot sizes x_i, $1 \le i \le n$, or in terms of the ratio of sublot sizes z_i, $1 \le i \le n$-1. Hereinafter , the problem is discussed in terms of z_i, $1 \le i \le n-1$ since, as shown by Theorem 3.14, monotonocity of the makespan function defined in the Z space becomes extremely helpful when developing a polynomial algorithm for the m x n lot streaming problem.

Theorem 3.13. *There is a one to one correspondence between the domains X and Z where*

$$X = \left\{(x_1, x_2, \ldots, x_n)^T \mid x_1 + x_2 + \ldots + x_n = 1 \text{ and } 0 < x_i < 1, \forall i\right\}$$

and

$$Z = \left\{(z_1, z_2, \ldots, z_{n-1})^T \mid 0 < z_i, \forall i\right\}$$

where the transformation from X to Z is $z_i = x_i / x_{i+1}$, $i = 1, 2, \ldots, n-1$.

Proof: For any $\mathbf{x} \in X$, the following system of equations gives the corresponding $\mathbf{z}$ in Z,

$$z_i x_{i+1} = x_i \quad \text{for } i = 1, 2, \ldots, n-1. \tag{3.80}$$

This system can be expressed as

$$X\mathbf{z} = \mathbf{x}'$$

where

$$\mathbf{x}' = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \end{pmatrix},$$

$$\mathbf{z} = \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{pmatrix}$$

and

$$X = \begin{pmatrix} x_2 & 0 & \ldots & 0 \\ 0 & x_3 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & 0 & x_n \end{pmatrix}$$

or X is the n-1 x n-1 diagonal matrix with x_{i+1} as the element in the ith row and ith column and zeroes elsewhere. As easily seen, $\det(X) = \prod_{i=2}^{n} x_i$. Since $x_i \neq 0$, $\det(X) \neq 0$. Therefore, X is a nonsingular matrix and for every $\mathbf{x}$, and thus, for every $\mathbf{x}'$, there exists a distinct $\mathbf{z}$. Also, since $x_i \neq 0$, $1 \leq i \leq n$, by Equation 3.80, $z_i \neq 0$, $1 \leq i \leq n$-1. Then, for every $\mathbf{x}$, $\mathbf{x} \in X$, not only is there a distinct $\mathbf{z}$, but $\mathbf{z} \in Z$.

Likewise, for any $\mathbf{z} \in Z$, the following system of equations gives the corresponding $\mathbf{x}$ in X,

$$\sum_{i=1}^{n} x_i = 1$$

$$x_i - z_i x_{i+1} = 0,\ i = 1,\ldots,n-1$$

This system is expressed as

$$\boldsymbol{Z}\,\mathbf{x} = \mathbf{e}_1$$

where

$$\boldsymbol{Z} = \begin{pmatrix} 1 & 1 & \ldots & 1 & 1 \\ 1 & -z_1 & 0 & \ldots & 0 \\ 0 & 1 & -z_2 & 0 & \ldots \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & 0 & 1 & -z_{n-1} \end{pmatrix}$$

or Z is the n x n matrix with all ones in its first row, z_{i-1} as the element in the ith row and ith column for $i = 2, 3, \ldots, n$, and ones as the elements in the ith row and $(i\text{-}1)$th column for $i = 2, 3, \ldots, n$ and zeroes elsewhere and $\mathbf{e}_1$ is the n x 1 vector with one as its first element and zeroes elsewhere. Expanding along the first row and simplifying, $\det(Z) = (-1)^{n-1} \sum_{k=1}^{n-1} \prod_{i=k}^{n-1} z_i + 1$. Since, $z_i \neq 0$, $\det(\boldsymbol{Z}) \neq 0$. Therefore, $\boldsymbol{Z}$ is a nonsingular matrix and for every $\mathbf{z} \in Z$ there exists a distinct $\mathbf{x}$. Also, since $z_i \neq 0$, the sublot sizes are also nonzero or $x_i \neq 0$. Then, for every distinct $\mathbf{z}$, $\mathbf{z} \in Z$, there is a distinct $\mathbf{x}$, $\mathbf{x} \in X$. Therefore, there is a one to one correspondence between the domains Z and X. □

Now, since

$$x_i = \frac{\prod_{j=i}^{n-1} z_j}{\sum_{k=1}^{n-1} \prod_{j=k}^{n-1} z_j + 1},\ i = 1,2,\ldots,n-1,$$

and

$$x_n = \frac{1}{\sum_{k=1}^{n-1}\prod_{j=k}^{n-1} z_j + 1}$$

the expression for the length of the path defined by feasible indices set I, in terms of z_i's and denoted $M_I(Z)$, is

$$M_I(Z) = \frac{\sum_{i=1}^{n-1}\prod_{j=i}^{n-1} a_i z_j}{\sum_{k=1}^{n-1}\prod_{j=k}^{n-1} z_j + 1} + T(I[n-1], m),$$

where $a_i = T(I[i\text{-}1], I[i]) - T(I[n\text{-}1], m)$ and $I[0] = 1$.

The expression for the makespan in terms of $M_I(Z)$ is

$$M_z(Z) = \max_{I \in K} M_I(Z)$$

and is defined over the domain

$$Z = \left\{ \mathbf{z} \in R^{n-1} \mid 0 < z_i, \forall i \right\}.$$

As is evident, the function for the makespan in terms of z_i's, $M_z(Z)$, is the upper envelope of the intersection of the functions $M_I(Z)$, $I \in K$. Finally, the optimal makespan M^* is

$$M^* = \min_{\mathbf{z} \in Z} M_z(Z).$$

In order to minimize $M_z(Z)$, or the makespan, the function $M_I(Z)$ must be characterized. As shown by Theorem 3.14 given below, $M_I(Z)$ is a monotone function in the direction of every element of Z.

Theorem 3.14. *The function $M_I(Z)$ is a monotone function in every element of Z.*

Proof: The partial derivative of $M_I(Z)$ with respect to z_j is

$$\frac{\partial M_I(Z)}{\partial z_j} = \frac{\sum_{i=1}^{j}\sum_{k=j+1}^{n-1}\prod_{l=i}^{j-1}\prod_{t=j+1}^{n-1}\prod_{r=k}^{n-1}(a_i - a_k)z_l z_t z_r + \sum_{i=1}^{j}\prod_{l=i}^{j-1}\prod_{t=j+1}^{n-1}a_i z_l z_t}{\left(\sum_{i=1}^{n-1}\prod_{k=i}^{n-1}z_k + 1\right)^2} \quad \text{for } j = 1,2,\ldots,n-2,$$

and

$$\frac{\partial M_I(Z)}{\partial z_j} = \frac{\sum_{i=1}^{j-1}\prod_{l=i}^{j-1}a_i z_l + a_{n-1}}{\left(\sum_{i=1}^{n-1}\prod_{k=i}^{n-1}z_k + 1\right)^2} \quad \text{for } j = n-1$$

where $a_i = T(I[i\text{-}1], I[i]) - T(I[n\text{-}1], m)$ and $I[0] = 1$.

The sign of $\partial M_I(Z)/\partial z_j$ is determined by its numerator which is invariant to z_j, $1 \le j \le n-1$. Thus, if z_j is increased, the sign of $\partial M_I(Z)/\partial z_j$ will not change. Hence, $M_I(Z)$ is a monotone function in the direction of z_j, $1 \le j \le n\text{-}1$. □

Now, as shown by Lemma 3.25, the minimum value of $M_I(Z)$, the length of the longest path defined by index set I, occurs at the upper or lower bounds of z_j, $1 \le j \le n\text{-}1$.

Lemma 3.25. *The minimum value of $M_I(Z)$ occurs at the lower and upper bounds of z_j, $1 \le j \le n-1$, as given in* Theorem 3.11.

Proof: Since $M_I(Z)$ is a monotone function in the directions of z_j, $1 \le j \le n\text{-}1$, depending on the signs of $\partial M_I(Z)/\partial z_j$, $1 \le j \le n-1$, the ratios of z_j's should be either increased or decreased to minimize $M_I(Z)$. If $\partial M_I(Z)/\partial z_j < 0$, increasing z_j decreases $M_I(Z)$, meanwhile, if $\partial M_I(Z)/\partial z_j > 0$, decreasing z_j also decreases the makespan, $M_I(Z)$. However, z_j's are bounded and cannot be decreased or increased indefinitely. As a result, z_j's must be decreased to its lowest level or increased to its highest level to minimize $M_I(Z)$. If $\partial M_I(Z)/\partial z_j = 0$, z_j can be increased (decreased) to its highest (lowest)

level without changing the value of $M_I(Z)$. Then, the minimum value of $M_I(Z)$, over the feasible ranges of z_j's, occurs at either the lower or upper bounds of z_j's. □

In Equation 3.65, z_j's are bounded by the ratios that are dependent on the crossover indices set I. Therefore, the values of z_j's that result in the path defined by feasible indices set $I[j]$ being the longest in the network are referred to as $z_j(I)$. Now, since $z_j(I)$'s are bounded, they cannot be increased or decreased indefinitely in order to reduce $M_I(Z)$ or, equivalently, the makespan. As shown by Lemma 3.25, given a longest path defined by indices set I, it is easy to find the values of $z_j(I)$'s that minimize this path. These values are either the lower or upper bounds of the ranges of $z_j(I)$'s. The bounds on $z_j(I)$ are determined by assuming that a critical box will be formed between the feasible index $I[j]$ and at least one of its adjacent feasible indices. However, although these values of $z_j(I)$'s give the minimized length of the path defined by the feasible indices set I, there is no guarantee that this minimized longest path is the minimal longest path in the network or, equivalently, the optimal makespan. Hence, a method of minimizing the makespan over all possible values of z_j's, and over all possible adjacent pairs of feasible indices is needed.

If every feasible index is evaluated to determine the set of feasible indices that yields the optimal makespan, this requires an exhaustive search considering the number of potentially critical paths in the network. Therefore, a more effective method, which requires fewer feasible index sets in the evaluation, is needed. Theorems 3.12, 3.13 and 3.14 help us in designing such an algorithm that does not need to perform an exhaustive search for all sets of feasible indices.

Theorem 3.12 provides that for every subpath of a critical path there is an alternate subpath $SP_j(i)$ of equal length that also traverses from node $(I[j\text{-}1], j\text{-}1)$ to node $(I[j+1], j+1)$. Since both subpaths are equal in length and have the same starting and ending nodes, the length of the path defined by the set of indices

$$\{I[1], \ldots, I[j\text{-}1], i, I[j+1], \ldots, I[n\text{-}1]\}$$

that follows the same path as $P(I)$ with the exception that this path follows the subpath $SP_j(i)$ instead of $SP_j(I[j])$ must be equal to the length of $P(I)$. While the crossover from column j to column $j+1$ takes place at node (i, j) in subpath $SP_j(i)$, this occurs at node $(I[j], j)$ in subpath $SP_j(I[j])$. As a result, for each critical path, there are at least two distinct crossover indices at column j, $1 \le j \le n\text{-}1$. Let us denote the first one as $I'[j]$ and the second one as $I[j]$. Since the subpaths they define are equal in length, $I[j]$ and $I'[j]$ are adjacent feasible indices. Therefore, all properties of $I[j]$ also pertain to $I'[j]$. Particularly, some z_j must exist such that

$$\frac{T(i', I'[j]-1)}{T(i'+1, I'[j])} \le z_j \le \frac{T(I'[j], i-1)}{T(I'[j]+1, i)} \quad \text{for } i' < I'[j] < i \text{ and } I'[j] \neq I[j].$$

Then, for every $I[j] \in I$ that defines a critical path, there is another index $I'[j]$, $I[j-1] \le I'[j] \le I[j+1]$ and $I'[j] \neq I[j]$, that can replace $I[j]$ and still give an alternate path of equal length. Therefore, for every critical path, there are at least $(n\text{-}1)$ and at most $(2^{n-1} - 1)$ alternate critical paths of equal length in the network. Hence, the bounds on z_j are

$$\frac{T(i', I[j]-1)}{T(i'+1, I[j])} \le z_j \le \frac{T(I[j], i-1)}{T(I[j]+1, i)}$$

and

$$\frac{T(i', I'[j]-1)}{T(i'+1, I'[j])} \le z_j \le \frac{T(I'[j], i-1)}{T(I'[j]+1, i)}.$$

Since $I'[j] < I[j]$ and $I'[j]$ is an adjacent index to $I[j]$,

$$z_j = \frac{T(I'[j], I[j]-1)}{T(I'[j]+1, I[j])}.$$

At this point, given a longest path, a method of obtaining a shorter longest path is needed. Let $P(I)$ be a longest path defined by the set I and $P'(I)$ be a longest path defined by alternative crossover indices set I'. Additionally, let $P(I_j^1)$ be the longest path which is defined by the set of machine indices $I_j^1 = I'_j \bigcup I_{j+1}$ and $P(I_j^2)$ be another longest path which is defined by the set $I_j^2 = I'_{j-1} \bigcup I_j$ where

$$I'_j = \{I'[1], \ldots, I'[j] \mid I'[k] \le I'[k+1], k = 1, \ldots, j-1\}$$

and

$$I_j = \{I[j], \ldots, I[n-1] \mid I[k] \le I[k+1], k = j, \ldots, n-2\}.$$

The lengths of all these paths are equal since each path is obtained by replacing any crossover index with its alternate crossover index. Hence, the length of $P(I_j^1)$ is

$$L_{P(I_j^1)} = \sum_{k=1}^{j} a'_k x_k + c_j x_{j+1} + \sum_{k=j+2}^{n-1} a_k x_k + T(I[n-1], m), \quad \text{for } 1 \le j < n\text{-}1,$$

and

$$L_{P(I_j^1)} = \sum_{k=1}^{n-1} b'_k x_k + T(I'[n-1], m), \quad \text{for } j = n\text{-}1,$$

where $a'_k = T(I'[k-1], I'[k]) - T(I[n-1], m)$, $a_k = T(I[k-1], I[k]) - T(I[n-1], m)$, $b'_k = T(I'[k-1], I'[k]) - T(I'[n-1], m)$, and $c_k = T(I'[k], I[k+1] - T(I[n-1], m)$.

However, the length of $P(I_j^2)$ is

$$L_{P(I_j^2)} = \sum_{k=1}^{j-1} a_k' x_k + c_{j-1} x_j + \sum_{k=j+1}^{n-1} a_k x_k + T(I[n-1],\, m), \quad \text{for } 1 \le j \le n\text{-}1.$$

Following these definitions, the partial derivatives of $P(I_j^1)$ and $P(I_j^2)$ with respect to z_j are

$$\frac{\partial L_{P(I_j^1)}}{\partial z_j} = \frac{1}{z_j}\left[\sum_{i=1}^{j}\left(a_i' x_i x_n + (a_i' - c_j) x_i x_{j+1} + \sum_{k=j+2}^{n-1}(a_i' - a_k) x_i x_k\right)\right], \quad \text{for } 1 \le j < n\text{-}1$$

and

$$\frac{\partial L_{P(I_j^1)}}{\partial z_j} = \frac{1}{z_j}\left[\sum_{i=1}^{n-1} b_i' x_i x_n\right], \quad \text{for } j = n\text{-}1$$

and

$$\frac{\partial L_{P(I_j^2)}}{\partial z_j} = \frac{1}{z_j}\left[\sum_{i=1}^{j}\left(c_{i-1} x_i x_n + \sum_{k=j+1}^{n-1}(c_{i-1} - a_k) x_i x_k\right)\right], \quad \text{for } j = 1,$$

$$\frac{\partial L_{P(I_j^2)}}{\partial z_j} = \frac{1}{z_j}\left[\sum_{i=1}^{j-1}\left(a_i' x_i x_n + \sum_{k=j+1}^{n-1}(a_i' - a_k) x_i x_k\right) + \sum_{k=j+1}^{n-1}(c_{j-1} - a_k) x_j x_k + c_{j-1} x_j x_n\right]$$

for $1 < j < n\text{-}1$

and

$$\frac{\partial L_{P(I_j^2)}}{\partial z_j} = \frac{1}{z_j}\left[\sum_{i=1}^{j-1} a_i' x_i x_n + c_{j-1} x_j x_n\right], \quad \text{for } j = n\text{-}1,$$

respectively.

These partial derivatives indicate whether or not the makespan defined by these paths can be decreased by changing z_j. For example, if $\partial L_{P(I_j^1)} / \partial z_j > 0$, z_j should be decreased in order to reduce the makespan. However, z_j is bounded as

$$\frac{T(i, I[j]-1)}{T(i+1, I[j])} \le z_j \le \frac{T(I[j], i'-1)}{T(I[j]+1, i')}$$

with

$$z_j = \frac{T(I'[j], I[j]-1)}{T(I'[j]+1, I[j])}$$

because $I[j]$ and $I'[j]$ are adjacent feasible indices. Therefore, if z_j is decreased to reduce the makespan, the path defined by $I[j]$ will no longer be a longest path in the network. The new value of z_j will be determined by $z_j = \max_{i<I'[j]}\left(\frac{T(i, I'[j]-1)}{T(i+1, I'[j])}\right)$. Following this, $I'[j]$ will be named as the new $I[j]$ while index i is labeled as the new $I'[j]$. However, if $\partial L_{P(I_j^1)} / \partial z_j < 0$, z_j should be increased in order to reduce the makespan. In this case, the path defined by $I'[j]$ will not be a longest path. The new value of z_j will be determined by $z_j = \min_{i>I[j]}\left(\frac{T(I[j], i-1)}{T(I[j]+1, i)}\right)$. Following this, $I[j]$ will be named as the new $I'[j]$ while index i is labeled as the new $I[j]$. Similarly, if $\partial L_{P(I_j^2)} / \partial z_j > 0$, z_j will be decreased to $z_j = \max_{i<I'[j]}\left(\frac{T(i, I'[j]-1)}{T(i+1, I'[j])}\right)$ while the crossover indices are updated as $I[j] = I'[j]$ and $I'[j] = i$. If $\partial L_{P(I_j^2)} / \partial z_j < 0$, z_j will be increased to $z_j = \min_{i>I[j]}\left(\frac{T(I[j], i-1)}{T(I[j]+1, i)}\right)$ while the crossover indices are updated as $I'[j] = I[j]$ and $I[j] = i$.

The above discussion can be summarized as follows,

- If $\partial L_{P(I_j^1)}/\partial z_j \le 0$ and $\partial L_{P(I_j^2)}/\partial z_j < 0$,

  or $\partial L_{P(I_j^1)}/\partial z_j < 0$ and $\partial L_{P(I_j^2)}/\partial z_j \le 0$, increase z_j.

- If $\partial L_{P(I_j^1)}/\partial z_j \ge 0$ and $\partial L_{P(I_j^2)}/\partial z_j > 0$,

  or $\partial L_{P(I_j^1)}/\partial z_j > 0$ and $\partial L_{P(I_j^2)}/\partial z_j \ge 0$, decrease z_j.

- If $\partial L_{P(I_j^1)}/\partial z_j \le 0$ and $\partial L_{P(I_j^2)}/\partial z_j > 0$,

  or $\partial L_{P(I_j^1)}/\partial z_j < 0$ and $\partial L_{P(I_j^2)}/\partial z_j \ge 0$, do nothing.

Given an initial set of crossover indices I, the rules above can be used to find another set of crossover indices that defines a shorter longest path. If $\partial L_{P(I_j^1)}/\partial z_j \le 0$ and $\partial L_{P(I_j^2)}/\partial z_j > 0$, or $\partial L_{P(I_j^1)}/\partial z_j < 0$ and $\partial L_{P(I_j^2)}/\partial z_j \ge 0$, the crossover index $I[j]$ cannot be updated since the first term shows that z_j should be increased while the second terms indicates that z_j should be decreased. When this is the case for all z_j's, a local minimum has been obtained for the lot streaming problem. However, a global minimum is needed to solve the problem optimally.

Global optimality is established in the following way. Theorem 3.13 shows that for every vector **z** of ratios of consecutive sublot sizes, there exists a unique vector **x** of sublot sizes and vice-versa. If $M_z(\mathbf{z})$ is the makespan of the problem in terms of the **z** vector, then $M_x(\mathbf{x}) = M_z(\mathbf{z})$. Furthermore, Theorem 3.15 below shows that a local minimum of $M_x(\mathbf{x})$ is also a local minimum to $M_z(\mathbf{z})$ and vice-versa. Meanwhile, since $M_x(\mathbf{x})$ is a convex function which is being minimized over a convex region X, a local minimum of $M_x(\mathbf{x})$ is a global minimum. Therefore, a local minimum of $M_z(\mathbf{z})$ is also a

global minimum. As a result, when a local minimum of $M_z(\mathbf{z})$ has been found, the optimal solution to the m x n lot streaming problem has been obtained.

Theorem 3.15. *The vector* $\mathbf{z}^*$ *of sublot size ratios is a local minima to* $M_z(\mathbf{z})$ *if and only if the corresponding vector* $\mathbf{x}^*$ *of sublot sizes is a local minima to* $M_x(\mathbf{x})$.

Proof: Let $\mathbf{x}^*$ be a local minima of $M_x(\mathbf{x})$. Then there exists an epsilon neighborhood of $\mathbf{x}^*$, $N_\varepsilon(\mathbf{x}^*)$, such that for every $\mathbf{x} \in N_\varepsilon(\mathbf{x}^*)$, $M_x(\mathbf{x}) \geq M_x(\mathbf{x}^*)$, where

$$N_\varepsilon(\mathbf{x}^*) = \left\{ \mathbf{x} \;\middle|\; |x_i - x_i^*| \leq \varepsilon_i \text{ and } \varepsilon_i > 0 \text{ for } i = 1,2,\ldots,n \right\}.$$

For any $\mathbf{x}$, $\mathbf{z}$ exists such that $z_i = x_i / x_{i+1}$, $i = 1,2,\ldots,n$. Since for any $\mathbf{x} \in N_\varepsilon(\mathbf{x}^*)$, $|x_i - x_i^*| < \varepsilon_i$ for $i = 1,2, \ldots, n$,

$$x_{i+1}^* - \varepsilon_{i+1} < x_{i+1} < x_{i+1}^* + \varepsilon_{i+1}.$$

Then, the following bounds are established on the reciprocal of x_{i+1},

$$\frac{1}{x_{i+1}^* + \varepsilon_{i+1}} < \frac{1}{x_{i+1}} < \frac{1}{x_{i+1}^* - \varepsilon_{i+1}}.$$

The bounds on the ratio of consecutive sublot sizes are

$$\frac{x_i^* - \varepsilon_i}{x_{i+1}^* + \varepsilon_{i+1}} < \frac{x_i}{x_{i+1}} < \frac{x_i^* + \varepsilon_i}{x_{i+1}^* - \varepsilon_{i+1}}$$

or

$$\frac{x_i^* - \varepsilon_i}{x_{i+1}^* + \varepsilon_{i+1}} < z_i < \frac{x_i^* + \varepsilon_i}{x_{i+1}^* - \varepsilon_{i+1}} \tag{3.81}$$

Therefore, for every $\mathbf{x} \in N_\varepsilon(\mathbf{x}^*)$, a vector $\mathbf{z}$ exists with each element z_i bounded as given above in Equation 3.81. By Theorem 3.13, this $\mathbf{z}$ is unique for every $\mathbf{x}$. Then, since $\varepsilon > 0$ and $x_i^* > 0$, ε_i' exists such that

$$\varepsilon_i' = \frac{\varepsilon_i}{x_i} > 0.$$

When $\varepsilon_i' x_i^*$ is used for ε_i in Equation 3.81, the following equation is obtained,

$$\left(\frac{x_i^*}{x_{i+1}^*}\right)\left(\frac{1-\varepsilon_i}{1+\varepsilon_{i+1}'}\right) < z_i < \left(\frac{x_i^*}{x_{i+1}^*}\right)\left(\frac{1+\varepsilon_i}{1-\varepsilon_{i+1}'}\right)$$

or

$$z_i^*\left(\frac{1-\varepsilon_i}{1+\varepsilon_{i+1}'}\right) < z_i < z_i^*\left(\frac{1+\varepsilon_i}{1-\varepsilon_{i+1}'}\right).$$

This yields the following,

$$z_i^*\left(1-\frac{\varepsilon_{i+1}'+\varepsilon_i}{1+\varepsilon_{i+1}'}\right) < z_i < z_i^*\left(1+\frac{\varepsilon_{i+1}'+\varepsilon_i}{1-\varepsilon_{i+1}'}\right)$$

or

$$-z_i^*\left(\frac{\varepsilon_{i+1}'+\varepsilon_i}{1+\varepsilon_{i+1}'}\right) < z_i - z_i^* < z_i^*\left(\frac{\varepsilon_{i+1}'+\varepsilon_i}{1-\varepsilon_{i+1}'}\right). \tag{3.82}$$

Therefore, for every $\mathbf{x} \in N_\varepsilon(\mathbf{x}^*)$, a unique vector $\mathbf{z}$ exists with each element z_i bounded as given above in Equation 3.82. Given this, a set of $\mathbf{z}$'s exists such that

$$-z_i^*\left(\frac{\varepsilon_{i+1}'+\varepsilon_i}{1+\varepsilon_{i+1}'}\right) < z_i - z_i^* < z_i^*\left(\frac{\varepsilon_{i+1}'+\varepsilon_i}{1+\varepsilon_{i+1}'}\right).$$

Now let

$$\delta_i = z_i^*\left(\frac{\varepsilon_{i+1}'+\varepsilon_i}{1+\varepsilon_{i+1}'}\right) > 0.$$

Then by simplifying the above bounds,

$$-\delta_i < z_i - z_i^* < \delta_i$$

or

$$\left| z_i - z_i^* \right| \le \delta_1,$$

$i = 1, 2, \ldots, n\text{-}1$, for each z_i in this set of $\mathbf{z}$'s. This set is $N_\delta(\mathbf{z}^*)$, a delta neighborhood of $\mathbf{z}^*$, and $\mathbf{z}^* \in N_\delta(\mathbf{z}^*)$. Since $M_z(\mathbf{z}) = M_x(\mathbf{x})$ for all $\mathbf{x}$ and $\mathbf{z}$ and $M_x(\mathbf{x}) \ge M_x(\mathbf{x}^*)$ for all $\mathbf{x} \in N_\varepsilon(\mathbf{x}^*)$, $M_z(\mathbf{z}) \ge M_z(\mathbf{z}^*)$ for every $\mathbf{z}^* \in N_\delta(\mathbf{z}^*)$. Therefore, $\mathbf{z}^*$ is a local minima of $M_z(\mathbf{z})$. The similar argument applies if $\mathbf{z}^*$ is a local minima of $M_z(\mathbf{z})$. □

After the necessary groundwork presented above, the algorithm that was developed for the m-machine n-sublot lot streaming problem is the following.

m x n Search Algorithm

0. Perform Feasible Index Search Algorithm to obtain the set of feasible indices, F, where $r = |F|$, $r \le m$, is the number of feasible indices and $F[i]$, $i = 1, \ldots, r$, represents the elements of the set F.

   Form $\mathbf{I}$ and $\mathbf{I}'$, two alternate crossover indices sets, by initially setting as $I'[j] = 1$ and $I[j] = F[2]$, for $j = 1, 2, \ldots, n\text{-}1$.

1. Set $z_j = \dfrac{T(I'[j], I[j]-1)}{T(I'[j]+1, I[j])}$ for $j = 1, 2, \ldots, n\text{-}1$.

   Make the following initializations. Start with the first pair of columns, i.e., $f = 1$, set number of fixed columns to zero, i.e., FC = 0, and set the rightmost pair of columns to $n-1$, i.e., UL = n-1.

2. DO WHILE (FC ≠ n-1)

   BEGIN {*main cycle*}

      DO WHILE ($f \le n$-1)

BEGIN {*traverse columns* 1 to *n*-1 *to check partial derivatives*}

IF $\frac{\partial L_{P(I_f^1)}}{\partial z_f} \le 0$ and $\frac{\partial L_{P(I_f^2)}}{\partial z_f} < 0$, or $\frac{\partial L_{P(I_f^1)}}{\partial z_f} < 0$ and $\frac{\partial L_{P(I_f^2)}}{\partial z_f} \le 0$,

BEGIN

i = UL

DO WHILE ($i \ge f$)

BEGIN {*increase the critical indices starting right hand side*}

Increase $I'[i]$ and $I[i]$ to the next feasible indices,

Set $z_j = \frac{T(I'[j], I[j]-1)}{T(I'[j]+1, I[j])}$, $1 \le j \le n-1$.

IF $\frac{\partial L_{P(I_f^1)}}{\partial z_f} \le 0$ and $\frac{\partial L_{P(I_f^2)}}{\partial z_f} > 0$, OR

$\frac{\partial L_{P(I_f^1)}}{\partial z_f} < 0$ and $\frac{\partial L_{P(I_f^2)}}{\partial z_f} \ge 0$,

BEGIN

UL = i - 1

IF ($i = f$)

UL = n-1

$i = f$-1 {to break the loop}

$f = f + 1$

END-IF

ELSE

BEGIN

IF ($i = f$)

UL = n-1

$i = i$-1

END-ELSE

END-DO {*increase the critical indices starting right hand side*}

END-IF

ELSE

BEGIN

IF $\frac{\partial L_{P(I_f^1)}}{\partial z_f} \geq 0$ and $\frac{\partial L_{P(I_f^2)}}{\partial z_f} > 0$, or $\frac{\partial L_{P(I_f^1)}}{\partial z_f} > 0$ and $\frac{\partial L_{P(I_f^2)}}{\partial z_f} \geq 0$,

BEGIN

$i = f$

DO WHILE ($i \leq n$-1)

BEGIN {*decrease the critical indices starting left hand side*}

Decrease $I'[j]$ and $I[j]$ to the previous feasible indices,

Set $z_j = \frac{T(I'[j], I[j]-1)}{T(I'[j]+1, I[j])}$, $1 \leq j \leq n-1$.

IF $\frac{\partial L_{P(I_f^1)}}{\partial z_f} \leq 0$ and $\frac{\partial L_{P(I_f^2)}}{\partial z_f} > 0$, OR

$\frac{\partial L_{P(I_f^1)}}{\partial z_f} < 0$ and $\frac{\partial L_{P(I_f^2)}}{\partial z_f} \geq 0$,

BEGIN

$i = n$ {to break the loop}

$f = f + 1$

ENDIF

ELSE

$i = i + 1$

END-DO {*decrease the critical indices starting left hand side*}

END-IF

ELSE

BEGIN

$f = f + 1$

UL = n-1

END-ELSE

END-ELSE

END-DO {*traverse columns* 1 to n *to check partial derivatives*}

DO FOR i = 1 to n-1 {*check the optimality condition for each column*}

BEGIN

IF $\frac{\partial L_{P(I_f^1)}}{\partial z_f} \leq 0$ and $\frac{\partial L_{P(I_f^2)}}{\partial z_f} > 0$, or $\frac{\partial L_{P(I_f^1)}}{\partial z_f} < 0$ and $\frac{\partial L_{P(I_f^2)}}{\partial z_f} \geq 0$

FC = FC +1

END-DO {*check if the optimality condition is satisfied for each column*}

IF (FC $\neq$ n-1) {*optimality is not achieved, repeat the main cycle once more*}

BEGIN

FC = 0

f = 1

END-IF

END-DO {*main cycle*}

3. The sublot sizes providing the best makespan is obtained via the critical indices.

Compute **z**, sublot size ratios vector, using critical machine indices sets I and I',

i.e. for $i = 1$ to n-1

$$z_i^* = \frac{T(I'[i], I[i]-1)}{T(I'[i]+1, I[i])}.$$

Calculate **x**, sublot sizes vector, using **z**.

i.e., for $i = 1$ to n

$$x_i^* = \frac{\prod_{j=i}^{n-1} z_j^*}{\sum_{k=1}^{n-1}\prod_{j=k}^{n-1} z_j^* + 1}.$$

Finally, the optimal makespan value is

$$M = T(1, I[1])x_1^* + \sum_{i=2}^{n-1} T(I[i-1], I[i])x_i^* + T(I[n-1], m)x_n^*.$$

The steps of the above algorithm will be explained in a little more detail in the following. Step 0 is essentially a preparation step which obtains the set of feasible machine indices, defines two sets of crossover indices and then initializes the elements of the sets to the first and second feasible indices, respectively. Step 1 continues with the initializations by setting all sublot size ratios to the value defined between the first and the second feasible indices. Therefore, initially all critical boxes in the network are situated between the first and the second feasible indices.

In the algorithm above, Step 2 handles all the computation in the algorithm. The *main cycle* runs until every pair of consecutive columns is handled, that is, the crossover indices that satisfy the optimality conditions are obtained for every sublot size ratio. In the process, the partial derivatives of the two alternative longest path functions for the sublot size ratio which is being studied are calculated. If both partial derivatives are negative for sublot size ratio z_i, then, there will be an improvement on the length of the longest path when the critical box between columns i and i+1 is moved down, that is, when the sublot size z_i is increased. However, if all critical boxes are at the same level, the critical boxes on the right hand side should be moved down first so that the critical box under study can be moved down to the next pair of feasible indices. Therefore, if this is the case, a block at the end of the row is moved down and the partial derivatives are updated. If there is still no change in the signs of the partial derivatives, the critical box on the left of the block just moved is moved down. If the critical box under study and all the boxes on the right hand side were moved once and the signs still do not satisfy the optimality, then the same process is repeated starting from the rightmost box. When the signs of the partial derivatives for the current sublot size ratio satisfy the rule, the crossover indices for the particular column are fixed and the next sublot size ratio is taken into consideration.

If both partial derivatives are found to have positive signs at the same time, the current box including the ones on the left are moved up one by one starting from the leftmost box since decreasing the particular sublot size ratio yields a shorter makespan. When the main step ends after finding the value for z_{n-1}, all the elements in the set of crossover indices is checked for optimality since having moved some boxes might have broken the optimality condition on some of the sublot size ratios that were already fixed.

If there is any column whose crossover indices generates partial derivatives with the same sign, the main step of the algorithm is applied once more. Although, it is very likely that the algorithm will find the optimal sublot sizes in one round; in case the main cycle is repeated, the number of computations will be much less since most of the columns already have their final crossover indices.

The complexity of the above algorithm is $O(n^3 * (m + n))$. This is due to the structure of the algorithm whose main body makes at most $O(m + n)$ iterations and during every iteration there is an inner body of $O(n)$ which calls for calculation of partial derivatives which requires $O(n^2)$ arithmetic operations. We rarely came across the examples that the first cycle of the main body did not locate the optimal solution. In those few cases, after the first cycle is completed, the optimality check called for another cycle, which located the optimal solution. The authors observed that the algorithm required a second main loop in only 11 problems out of 550 randomly generated lot streaming problems. The remaining problems have been optimally solved through single loops.

3.6.3 An $O(n^2 * (m + n))$ Algorithm for the m x n Problem

The algorithm that was presented above has been very efficient as the first polynomial-time alternative to linear programming when solving a general size lot streaming problem. However, since this particular algorithm requires frequent use of highly complicated partial derivatives, a simpler algorithm was desirable. Our research efforts yielded another algorithm on the general size lot streaming problems. The new algorithm, which is called "the m x n Block Algorithm", essentially uses the same search pattern in the network while looking for a shorter longest path as the previous algorithm.

The difference is that it simply compares the length of the current longest path with those of the new longest paths that can be obtained by moving a given crossover index to the next feasible index upward or downward in the network. When the movements in both directions on every column fail to yield a shorter longest path, it is said that the current set of crossover indices provides a local optimum. The discussion in the previous section on the global optimality of a local optimum obtained for the lot streaming problem is again valid for the new algorithm. The new algorithm for the m machine n sublot lot streaming problem is presented below.

m x n Block Algorithm

0. Obtain the set of feasible indices, F; let r, $r \leq m$, be the number of feasible indices.

1. Form **C**, crossover indices vector, by initially setting each element to the first feasible machine index, which is 1,

   i.e., set $C_i = 1$, for $i = 1$ to n-1.

   Compute **z**, sublot size ratios vector, using initial crossover indices,

   i.e., $z_i = \dfrac{T(1, F_2 - 1)}{T(2, F_2)}$, for $i = 1$ to n-1,

   where $T(i, j) = \sum_{k=i}^{j} p_k$, while p_k represents the unit processing time on machine k.

   Calculate **x**, sublot sizes vector, using **z**,

   i.e., $x_i = \dfrac{\prod_{j=i}^{n-1} z_j}{\sum_{k=1}^{n-1} \prod_{j=k}^{n-1} z_j + 1}$ for $i = 1$ to n.

Compute the makespan using **C**, F, and **x**,

i.e., $M = T(1, F_{C_1})x_1 + \sum_{i=2}^{n-1} T(F_{C_{i-1}}, F_{C_i})x_i + T(F_{C_{n-1}}, m)x_n$,

where F_i is the ith element of the feasible index set F.

Set the initial value of the minimal makespan to the value computed above,

i.e., MinMKS = M.

Set **CR**, the critical indices vector, to current crossover indices,

i.e., $CR_i = C_i$, for i = 1 to n-1.

Set ITER = 1, FXD = 1, UL = n-1.

2. DO WHILE (ITER < r-1) AND (FXD < n)

BEGIN {*main cycle*}

FOR i = UL DOWN TO i = FXD

BEGIN {*increase critical indices starting RHS*}

$C_i = C_i + 1$.

Compute **z**, sublot size ratios vector, using initial crossover indices,

i.e., $z_i = \dfrac{T(F_{C_i}, F_{C_i+1} - 1)}{T(F_{C_i} + 1, F_{C_i+1})}$, for i = 1 to n-1.

Calculate **x**, sublot sizes vector, using **z**.

Compute the makespan using **C**, F, and **x**.

IF $M \leq$ MinMKS

BEGIN

MinMKS = M

For i = 1 to n-1

$CR_i = C_i$

END-IF

ELSE

BEGIN

$C_i = C_i - 1$

FXD = FXD + 1

END-ELSE

END-FOR {*increase critical indices starting RHS*}

ITER = ITER + 1

UL = n - 1

DO-END {*main cycle*}

3. *Check for improvement*

FOR i = 1 TO n-1

$C_i = CR_i$

Define temporary indices $C_0 = 1$ and $C_n = F_{n-1}$.

FOR i = 1 TO n-1

BEGIN {*check for improvement at every column*}

IF ($C_i > C_{i-1}$)

BEGIN {*check upwards*}

$C_i = C_i - 1$

Compute **z**, **x** and the makespan.

IF $M \leq$ MinMKS

BEGIN {record the shorter longest path and new crossover indices}

MinMKS = M

For i = 1 to n-1

$CR_i = C_i$

END-IF {record the shorter longest path and new crossover indices}

ELSE

$C_i = C_i + 1$ {take the step back}

END-IF {*check upwards*}

IF ($C_i < C_{i+1}$)

BEGIN {*check downwards*}

$C_i = C_i + 1$

Compute **z**, **x** and the makespan.

IF $M \leq$ MinMKS

BEGIN {record the shorter longest path and new crossover indices}

MinMKS = M

For i = 1 to n-1

$CR_i = C_i$

END-IF {record the shorter longest path and new crossover indices}

ELSE

$C_i = C_i - 1$ {take the step back}

END-IF {*check downwards*}

END-FOR {*check for improvement at every column*}

4. The minimum makespan obtained by the heuristic is MinMKS.

The sublot sizes providing MinMKS value are

$$x_i^* = \frac{\prod_{j=i}^{n-1} z_j^*}{\sum_{k=1}^{n-1} \prod_{j=k}^{n-1} z_j^* + 1},$$

where **z** is the sublot size ratios vector obtained through **CR**, the vector of critical indices.

The complexity of the above algorithm is $O(n^2 * (m + n))$. This is due to the structure of the algorithm whose main body makes at most $O(m + n)$ iterations and during every iteration calculation of sublot sizes requires $O(n^2)$ arithmetic operations. There were some rare examples that the main cycle of the algorithm did not locate the optimal solution. Therefore, in order to increase the effectiveness of the algorithm, we inserted an optimality check in the third step after the main cycle. This step has $O(n^3)$ complexity, and therefore, it does not add to the overall complexity. The third step has been active in determining the set of crossover indices in only a small number of problems. In those few cases, after the main cycle has been completed, the third step simply moved one single crossover index to the previous feasible index, which located the optimal solution. The third step adjusted the final set of crossover indices in only 11 problems out of 550 randomly generated lot streaming problems. The remaining 539 sample problems have been solved without being affected by the third step.

CHAPTER 4
EXTENSIONS TO THE BASIC LOT STREAMING MODEL

The lot streaming model that was reviewed in the previous chapter can be considered as the basic model in single-job lot streaming area. The basic model have been studied by many researchers to some extent. However, this model has some limitations since it only considers an idealistic system. One of the idealistic points is that the production lot can be theoretically partitioned into an infinite number of sublots and therefore, sublot sizes are regarded as continuous variables. The basic model ignores setup activities on machines and transportation time between machines. Although the basic lot streaming model may provide insight on more complex environments, it is necessary to make some modifications on the lot streaming model to better represent the real manufacturing environments. Therefore, in this chapter, the basic lot streaming model is modified to include setup times on machines and transfer times between machines, which are not small enough to ignore in many manufacturing environments, and the effect of these changes on the solution schemes are also studied.

4.1 Lot Streaming Problems with Setups Explicitly Considered

As pointed out earlier, the basic lot streaming formulation does not explicitly consider any setup time in flow shop. However, setup times are important elements that need to be examined carefully for scheduling purposes. Poorly scheduled setups are very likely to increase the makespan of the production lot due to an unnecessary delay before

processing a sublot on a machine that is halted for a setup and thus, reduce the productivity of the production line. Therefore, lot streaming problems need to include setups as well.

The basic lot streaming problem formulation can be adapted to include setups. However, before making any attempts to remodel the problem, types of setups must be known. As stated in Chapter 1, setups are categorized as attached setups and detached setups, depending on whether a machine requires the arrival of some production items from the preceding machine or not, respectively. Even though, it may be possible to observe both types in the same manufacturing environment, only the pure cases with only one type of setups will be considered here in order to be accurate at this initiative. Another assumption is that setups on every machine are performed only once and before the processing of the first sublot. Once the setups are completed for the first sublot, the successive sublots do not incur any setup-related delay. The modifications on the basic lot streaming model concerning both types of setup activities will be examined separately in the following sections.

4.1.1 Lot Streaming Problems with Detached Setups

If the setups are detached, setups can be performed as soon as the machines becomes available without waiting for any work pieces to arrive. Baker's LP formulation [Bak87] will be modified to reflect the principle in handling the detached setups. The LP formulation modified for detached setups will basically check whether there is a delay because of the setups before processing the first sublot. In the revised formulation, unless the setup on a machine is longer than the time it takes for the first sublot to reach to the given machine, setups do not affect the schedule that was created without considering

setups. Therefore, the constraints (3.2) and (3.4) in the formulation (P3.1) are replaced by the following,

$$T_{11} \geq p_1 x_1 + S_1 \tag{4.1}$$

and

$$T_{i1} \geq \max\{T_{i-1,1}, S_i\} + p_i x_1, \qquad 2 \leq i \leq m, \tag{4.2}$$

$$T_{ij} \geq T_{i-1,j} + p_i x_j, \qquad 2 \leq i \leq m,\ 2 \leq j \leq n, \tag{4.3}$$

where S_i is the length of the setup time that must be performed on machine i before processing the first sublot. In order to represent (4.2) in an LP formulation, it must be broken up to the following two constraints,

$$T_{i1} \geq T_{i-1,1} + p_i x_1, \qquad 2 \leq i \leq m, \tag{4.4}$$

and

$$T_{i1} \geq S_i + p_i x_1, \qquad 2 \leq i \leq m. \tag{4.5}$$

After the modifications, the LP formulation for the lot streaming problem with detached setups is provided below.

(P4.1) minimize T_{mn}

subject to

$$x_1 + x_2 + \ldots + x_n = 1 \tag{4.6}$$

$$T_{11} \geq S_1 + p_1 x_1 \tag{4.7}$$

$$T_{ij} \geq T_{i,j-1} + p_i x_j, \qquad 1 \leq i \leq m,\ 2 \leq j \leq n \tag{4.8}$$

$$T_{i1} \geq T_{i-1,1} + p_i x_1, \qquad 2 \leq i \leq m, \tag{4.9}$$

$$T_{i1} \geq S_i + p_i x_1, \qquad 2 \leq i \leq m. \tag{4.10}$$

$$T_{ij} \geq T_{i-1,j} + p_i x_j, \qquad 2 \leq i \leq m,\ 2 \leq j \leq n, \tag{4.11}$$

$$T_{ij} \geq 0, \qquad 1 \leq i \leq m,\ 1 \leq j \leq n \qquad (4.12)$$

$$x_j \geq 0, \qquad 1 \leq j \leq n \qquad (4.13)$$

In this case, there will be (m-1) additional constraints. In order to represent the lot streaming problem with detached setups, an additional column of nodes whose weights are related to the setup time of each machine is introduced as shown in Figure 4.1 in the network representation originally developed by Baker and Pyke [Bak90]. In this network, column 0 denote the setup times associated with each machine through parameters S_i''s which are defined as follows,

$$S_1' = S_1,$$

and

$$S_i' = \max\left\{\left(S_i - \sum_{k=1}^{i-1} S_k'\right),\ 0\right\}, \quad i = 2,\ldots,m.$$

Therefore, $S_i \leq \sum_{k=1}^{i} S_k', \quad i = 1,\ldots,m.$ In the modified network, node (i, j), $1 \leq i \leq m$ and $1 \leq j \leq n$, still represents the processing of sublot j on machine i, which takes time px_j. For a given sublot size vector **x**, the length of the longest path in the network corresponds to the makespan of the lot streaming problem with detached setups. Since the objective function in the lot streaming problem is to minimize the makespan, the necessary action is to find the sublot sizes that yield the longest path with minimal length in the network representation. Therefore, the problem is transformed to minimization of the length of the longest path in the network.

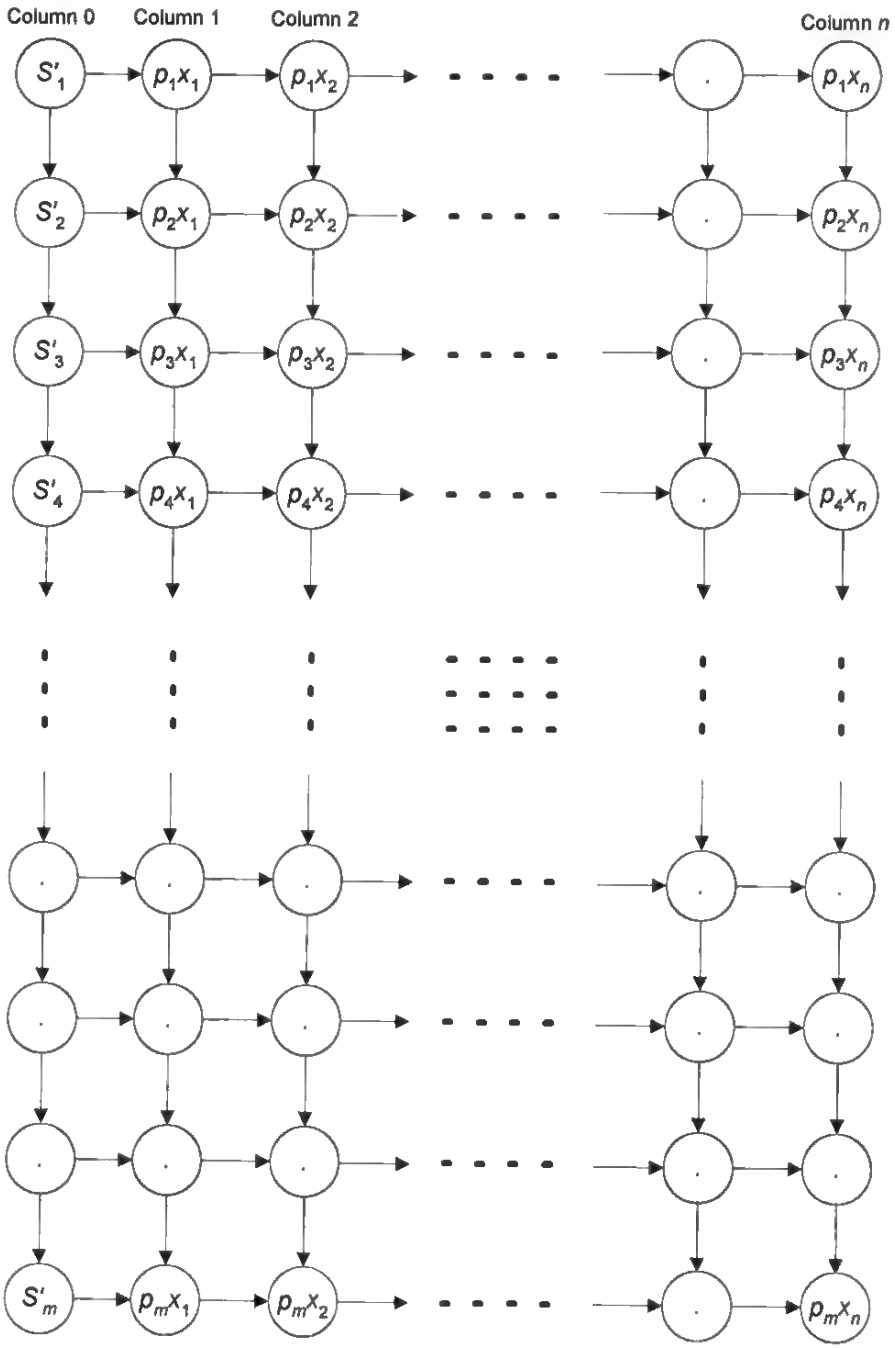

Figure 4.1 Network representation of the lot streaming problem with detached setups

4.1.1.1 Two-sublot problem with detached setups ($F, m \mid S, II, DS \mid 2, C, R \mid C_{max}$)

In order to obtain some insight on the general size lot streaming problem with detached setups, we will start the study with the 2 sublot problem. The LP formulation for the m machine 2 sublot problem with detached setups is given below.

(P4.2) minimize T_{m2}

subject to

$$x_1 + x_2 = 1 \tag{4.14}$$

$$T_{11} \geq p_1 x_1 + S_1 \tag{4.15}$$

$$T_{i2} \geq T_{i1} + p_i x_2, \quad 1 \leq i \leq m, \tag{4.16}$$

$$T_{i1} \geq T_{i-1,1} + p_i x_1, \quad 2 \leq i \leq m, \tag{4.17}$$

$$T_{i1} \geq S_i + p_i x_1, \quad 2 \leq i \leq m, \tag{4.18}$$

$$T_{i2} \geq T_{i-1,2} + p_i x_2, \quad 2 \leq i \leq m, \tag{4.19}$$

$$T_{ij} \geq 0, \quad 1 \leq i \leq m, j = 1, 2, \tag{4.20}$$

$$x_j \geq 0, \quad j = 1, 2. \tag{4.21}$$

The network representation corresponding to the above LP formulation of the 2 sublot problem with detached setups is also given in Figure 4.2. In the activity-on-node representation provided below, node $(i, 0)$, $1 \leq i \leq m$, represents the setup of machine i. The nodes (i, j), $1 \leq i \leq m$, $j = 1, 2$, represent the processing of sublot j, on machine i. Inside the nodes, both the node labels and node weights are provided. The constraint (4.15) corresponds to the horizontal precedence relationship between nodes (1, 0) and (1, 1) while the constraint set (4.18) denotes the horizontal precedence relationships between the remaining nodes of column 0 and column 1. The horizontal precedence relationships between the nodes of column 1 and column 2 are represented by the

constraint set (4.16). The constraint sets (4.17) and (4.19) state the vertical precedence relationships on column 1 and column 2, respectively. The makespan of the lot streaming problem corresponds to the longest path between the nodes (1, 0) and (m, 2) for the specified values of x_1 and x_2.

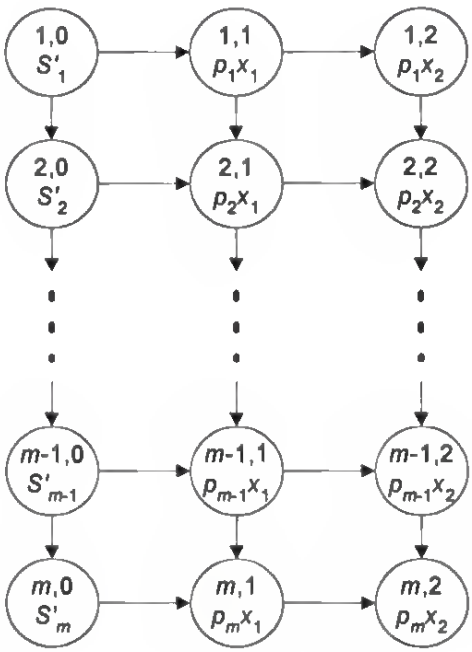

Figure 4.2 Network representation of m x 2 problem with detached setups

Definition 4.1. *In the network representation of the two sublot problem with detached setups, a path emanating from node* (1, 0) *and reaching to node* (m, 2) *can be represented by two machine indices,* i_1 and i_2, $1 \le i_1 \le i_2 \le m$, *at which the path crosses over from column* 0 *to column* 1 *and from column* 1 *to column* 2, *respectively.*

According to Definition 4.1, the length of an arbitrary path connecting the nodes (1, 0) and (m, 2) and crossing over from column 0 to column 1 at machine i_1 and crossing

over from column 1 to column 2 at machine i_2 can be denoted by $M_{i_1 i_2}(x_1, x_2)$. The length of this path can be obtained from

$$M_{i_1 i_2}(x_1, x_2) = \sum_{k=1}^{i_1} S'_k + T(i_1, i_2)x_1 + T(i_2, m)x_2,$$

where $T(i, j) = \sum_{k=i}^{j} p_k$. Since the production lot is split into only two sublots, determining the size of one of the sublots automatically determines the size of the other one. Therefore, since $x_1 + x_2 = 1$, x_2 can be replaced by $1 - x_1$, and in the formulation x_1 can simply be denoted by x. After these modifications, the length of the path becomes a linear function of x as follows,

$$\begin{aligned} M_{i_1 i_2}(x) &= \sum_{k=1}^{i_1} S'_k + T(i_1, i_2)x + T(i_2, m)(1 - x) \\ &= \big(T(i_1, i_2 - 1) - T(i_2 + 1, m)\big)x + \sum_{k=1}^{i_1} S'_k + T(i_2, m) \\ &= c_{i_1 i_2} x + \sum_{k=1}^{i_1} S'_k + T(i_2, m) \end{aligned} \tag{4.22}$$

where $c_{i_1 i_2} = T(i_1, i_2 - 1) - T(i_2 + 1, m)$.

Since the makespan for specified sublot sizes corresponds to the longest path in the network, the makespan for the specific x value will be the following,

$$M(x) = \max_{(i_1, i_2) \in K} M_{i_1 i_2}(x), \tag{4.23}$$

where $K = \{(r, s) \mid 1 \le r \le s \le m\}$.

In the above formulation, $M(x)$ is defined in the domain, $X = \{x \in \Re \mid 0 \le x \le 1\}$, which is a convex region. Then, in order to find the optimal makespan, the function

denoting the length of the longest path should be minimized over the feasible domain as follows,

$$M^* = \min_{0 \le x \le 1} M(x). \tag{4.24}$$

The overall process can be outlined by the following min-max objective function,

$$M^* = \min_{0 \le x \le 1} \max_{(i_1, i_2) \in K} M(x). \tag{4.25}$$

The above min-max problem can also be transformed into a form of a linear program with two decision variables and $\left(m(m+1)/2\right)$ number of constraints as follows,

(P4.3) minimize M

subject to $$M - c_{i_1 i_2} x \ge \sum_{k=1}^{i_1} S'_k + T(i_2, m), \qquad (i_1, i_2) \in K \tag{4.26}$$

$$M \ge 0 \tag{4.27}$$

$$0 \le x \le 1 \tag{4.28}$$

where x is the size of the first sublot and M is the makespan.

As seen above, the two sublot problem with detached setups reduces to the minimization problem over a convex envelope created by the maximum of the linear functions associated with the pairs of crossover machine indices. At this point, Williams and Tüfekçi's solution technique for m machine 2 sublot problem that was introduced in Section 3.3.3 is applicable with minor modifications. Since the number of linear functions has risen to $\left(m(m+1)/2\right)$, modified algorithm will obtain the optimal solution in $O(m^2)$ number of steps.

4.1.1.2 Three-sublot problem with detached setups ($F, m \mid S, II, DS \mid 3, C, R \mid C_{max}$)

The LP formulation of the m machine 3 sublot lot streaming problem with detached setups is provided below in order to facilitate the discussion on this problem.

(P4.4) minimize T_{m3}

subject to

$$x_1 + x_2 + x_3 = 1 \tag{4.29}$$

$$T_{11} \geq p_1 x_1 + S_1 \tag{4.30}$$

$$T_{ij} \geq T_{i,j-1} + p_i x_j, \quad 1 \leq i \leq m, \ j = 2, 3, \tag{4.31}$$

$$T_{i1} \geq T_{i-1,1} + p_i x_1, \quad 2 \leq i \leq m, \tag{4.32}$$

$$T_{i1} \geq S_i + p_i x_1, \quad 2 \leq i \leq m, \tag{4.33}$$

$$T_{ij} \geq T_{i-1,j} + p_i x_j, \quad 2 \leq i \leq m, \ j = 2, 3, \tag{4.34}$$

$$T_{ij} \geq 0, \quad 1 \leq i \leq m, j = 1, 2, 3, \tag{4.35}$$

$$x_j \geq 0, \quad j = 1, 2, 3. \tag{4.36}$$

The project network representation corresponding to the m machine 3 sublot lot streaming problem with detached setups is presented in Figure 4.3 below.

Similar to the two sublot case, here again the length of the longest path in the network representation corresponds to the makespan for the specified combination of the sublot sizes. When searching for the minimum makespan, it is necessary to find the sublot sizes that result in the minimal longest path in the network.

Definition 4.2. *In the network representation of the three sublot problem with detached setups, a path emanating from node* (1, 0) *and reaching to node* (m, 3) *can be represented by three machine indices,* i_1, i_2 and i_3, $1 \leq i_1 \leq i_2 \leq i_3 \leq m$, *at which the path crosses over*

from column 0 *to column* 1, *from column* 1 *to column* 2 *and from column* 2 *to column* 3, *respectively.*

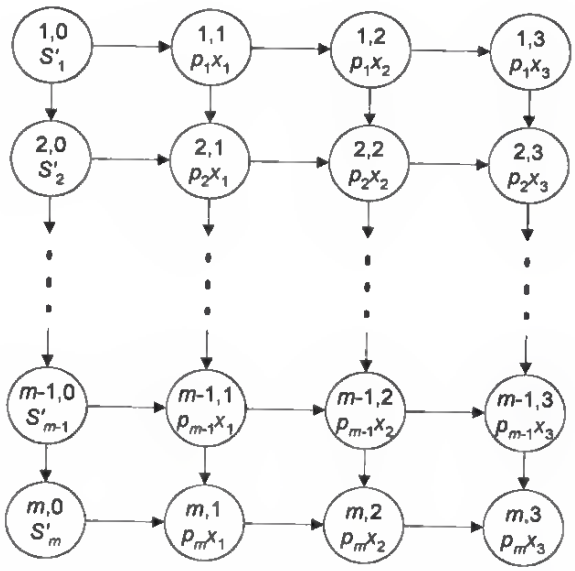

Figure 4.3 Network representation of m x 3 problem with detached setups

By Definition 4.2, the length of a path that traverses from the node (1, 0) to (m, 3) can be formulated by the following equation,

$$M_{i_1i_2i_3}(x_1,x_2,x_3) = \sum_{k=1}^{i_1} S'_k + T(i_1,i_2)x_1 + T(i_2,i_3)x_2 + T(i_3,m)x_3, \tag{4.37}$$

where $T(i,j) = \sum_{k=i}^{j} p_k$, while p_k is the unit processing time on machine k. The makespan function of the lot streaming problem with detached setups, $M_M(x_1,x_2,x_3)$, is defined as

$$M_M(x_1,x_2,x_3) = \max_{(i_1,i_2,i_3)\in K} M_{i_1i_2i_3}(x_1,x_2,x_3) \tag{4.38}$$

where $K = \{(r,s,t) \mid 1 \le r \le s \le t \le m\}$. Minimizing the makespan function over the domain $X = \{\mathbf{x} \in \Re^3 \mid x_1 + x_2 + x_3 = 1,\ 0 \le x_i \le 1,\ i = 1,\ 2,\ 3\}$, obtains the optimal solution to the problem as

$$M^* = \min_{\mathbf{x}\in X} M_M(x_1, x_2, x_3). \tag{4.39}$$

When the size of the third sublot, x_3, is assumed to be a fixed parameter, c, the length of the path given in Equation 4.37 will reduce to an expression with only one variable due to the constraint (4.29) in LP formulation P4.4, as seen below.

$$M_{i_1 i_2 i_3}(x_1, x_2, x_3) = \sum_{k=1}^{i_1} S'_k + T(i_1, i_2)x_1 + T(i_2, i_3)(1 - c - x_1) + T(i_3, m)c, \tag{4.40}$$

The following is obtained by arranging the terms,

$$M_{i_1 i_2 i_3}(x_1, c) = \big(T(i_1, i_2) - T(i_2, i_3)\big)x_1 + \big(T(i_3, m) - T(i_2, i_3)\big)c + \sum_{k=1}^{i_1} S'_k + T(i_2, i_3), \tag{4.41}$$

where x_3 is replaced by the constant c, and x_2 becomes $(1 - c - x_1)$.

The above transformation yields $\sum_{i=1}^{m} \frac{i(i+1)}{2}$ linear functions of x_1, which is $O(m^3)$, given x_3 is known. Since the makespan corresponds to the longest path in the network representation, the maximum of all resulting functions is needed,

$$M(x_1, c) = \max_{(i_1, i_2, i_3)\in K} M_{i_1 i_2 i_3}(x_1, c). \tag{4.42}$$

where $K = \{(r, s, t) \mid 1 \le r \le s \le t \le m\}$.

Finally, the minimal makespan for $x_3 = c$ is obtained by minimizing $M(x_1)$ over the domain $0 \le x_1 \le 1 - c$, where all the functions are defined,

$$M^*(c) = \min_{0\le x_1\le 1-c} \max_{(i_1, i_2, i_3)\in K} M_{i_1 i_2 i_3}(x_1). \tag{4.43}$$

Therefore, the solution process can also be expressed by the following LP formulation with 2 variables and $O(m^3)$ constraints,

(P4.5) min $M(c)$

$$\text{s.t.} \quad M(c) - \left(T(i_1,i_2) - T(i_2,i_3)\right)x_1 \geq \sum_{k=1}^{i_1} S'_k + T(i_2,i_3)(1-c) + T(i_3,m)c, \qquad (4.44)$$

$$M(c) \geq 0 \qquad (4.45)$$

$$0 \leq x_1 \leq 1 - c \qquad (4.46)$$

The linear program above can be solved by the search algorithm developed by Williams and Tüfekçi [Wil95] in $O(m^3)$ for a given c since there are $O(m^3)$ linear functions defined over the feasible domain. In order to obtain a solution close enough to the global optimal, the parameter c must also assume a proper value. Considering the increased number of linear functions that need to be included in the search for the minimal longest path, there is a need to restructure the above mentioned solution method specifically for the 3 sublot detached setup problem. The modified algorithm will be called as m x 3 Detached Setup Algorithm (m x 3 DSA). In the algorithm, we define $NUM(I)$ as the number of elements in the set I. Also, $\lfloor x \rfloor$ represents the largest integer smaller than or equal to x.

m x 3 Detached Setup Algorithm (for fixed x_3 value)

0. Let $L = \{l_1, l_2, \cdots, l_q\}$, where $q = \sum_{i=1}^{m} \frac{i(i+1)}{2}$, be the set of the functions defining the length of paths in the network representing the lot streaming problem with detached setups and $0 < c < 1$, where c is the parametric value of x_3, be fixed. Also let $\delta = 1/Q$, where Q is the number of items in the production lot.

Set $x_l = 0$, the lower limit for the first sublot size.

Set $x_u = 1 - c$, the upper limit for the first sublot size.

1. For each $t \in N = \left\{1,3,\ldots,2\left\lfloor \frac{NUM(L)}{2} \right\rfloor - 1\right\}$, set

$$x_{t,t+1} = \frac{R_{t+1} - R_t}{e_t - e_{t+1}}$$

where $R_t = \sum_{k=1}^{i_{1,t}} S'_k + T(i_{2,t}, i_{3,t})(1-c) + T(i_{3,t}, m)c,$

$e_t = \left(T(i_{1,t}, i_{2,t}) - T(i_{2,t}, i_{3,t})\right)$, $(i_{1,t}, i_{2,t}, i_{3,t}) \in K$, and $i_{1,t}, i_{2,t}$ and $i_{3,t}$ are cross-over indices associated with the path represented by the function l_t.

2. For each $t \in N$,

IF $x_{t,t+1} \geq x_u$,

Update N and L as $N\backslash\{t\}$ and $L\backslash\{k\}$, respectively, where

$$k = \begin{cases} l_t & \text{if } e_t > e_{t+1} \\ l_{t+1} & \text{otherwise.} \end{cases}$$

IF $x_{t,t+1} \leq x_l$,

Update N and L as $N\backslash\{t\}$ and $L\backslash\{k\}$, respectively, where

$$k = \begin{cases} l_t & \text{if } e_t < e_{t+1} \\ l_{t+1} & \text{otherwise.} \end{cases}$$

3. IF $NUM(L) = 2$,

Algorithm is complete, go to Step 7.

IF $N = \{\varnothing\}$,

Relabel $L = \{l_1, l_2, \ldots, l_{NUM(L)}\}$ and go to Step 1,

Set

$$\bar{x} = \sum_{t \in T} \frac{x_{t,t+1}}{NUM(T)}$$

4. Let $E = \{r \in L \mid e_r \bar{x} + R_r \geq e_l \bar{x} + R_l, \forall l \in L\}$.

   Set $slope1 = \min_{r \in E}\{e_r\}$, $slope1i = r$, and $slope2 = \max_{r \in E}\{e_r\}$, $slope2i = r$.

5. IF $(slope1)(slope2) \leq 0$,

   Set $x^* = \bar{x}$.

   Set $M^*(c) = \max_{l \in L}\{e_l \bar{x} + R_l\}$, stop.

6. IF $slope1 > 0$,

   Set $x_u = \bar{x}$ and for each $t \in N$ such that $x_{t,t+1} \geq \bar{x}$.

   Update L as $L\backslash\{k\}$ where

$$k = \begin{cases} l_t & \text{if } e_t > e_{t+1} \\ l_{t+1} & \text{otherwise.} \end{cases}$$

   IF $slope1 < 0$,

   Set $x_l = \bar{x}$ and for each $t \in N$ such that, $x_{t,t+1} \leq \bar{x}$.

   Update L as $L\backslash\{k\}$ where

$$k = \begin{cases} l_t & \text{if } e_t < e_{t+1} \\ l_{t+1} & \text{otherwise.} \end{cases}$$

7. Relabel $L = \{l_1, l_2, \ldots, l_{NUM(L)}\}$.

   IF $NUM(L) = 2$,

   Optimal size of sublot 1 is

$$x^* = \frac{R_2 - R_1}{e_1 - e_2}.$$

   Set $slope1 = \min_{r=1,2}\{e_r\}$, $slope1i = r$, and $slope2 = \max_{r=1,2}\{e_r\}$, $slope2i = r$.

   ELSE

Go to Step 1.

8a. The optimal makespan is $M^*(c) = e_1 x^* + R_1$.

8b. *Adjustment step*

IF $(x^* < x_l)$ OR $(x^* < \delta)$

IF $(\delta > x_l)$

$x^* = \delta$

ELSE

$x^* = x_l$

$M^*(c) = e_{slope2i} x^* + R_{slope2i}$.

IF $(x^* > x_u)$ AND $(x_u > \delta)$

$x^* = x_u$

$M^*(c) = e_{slope1i} x^* + R_{slope1i}$.

IF $(e_{slope1i} > 0)$ AND $(e_{slope2i} > 0)$

IF $(\delta > x_l)$

$x^* = \delta$

ELSE

$x^* = x_l$

$M^*(c) = e_{slope1i} x^* + R_{slope1i}$.

IF $(e_{slope1i} < 0)$ AND $(e_{slope2i} < 0)$

IF $(\delta > x_u)$

$x^* = \delta$

ELSE

$x^* = x_u$

$M^*(c) = e_{slope2i} x^* + R_{slope2i}$.

The adjustment step is added to eliminate some irregularities that are results of not working over the complete line of [0, 1], which the original method uses. The solution obtained from m x 3 Detached Setup Algorithm (DSA) is optimal only if the specified x_3 value, c, is in fact optimal. In order to be close enough to a globally optimal solution, this algorithm should be employed in a bisection framework as provided below.

Bisection Algorithm

INPUT $m, p_1, \ldots, p_m, I, \varepsilon$.

LET $l = 0$ and $u = 1$.

FOR $i = 1$ TO I

BEGIN {*perform I iterations to obtain best value for* x_3}

$c_m = (l + u) / 2$

$c_l = c_m - \varepsilon$

$c_r = c_m + \varepsilon$

$M_l = m$ x 3 DSA $(m, p_1, \ldots, p_m, S_1, \ldots, S_m, c_l)$

$M_r = m$ x 3 DSA $(m, p_1, \ldots, p_m, S_1, \ldots, S_m, c_r)$

IF $M_l < M_r$

$u = c_r$

ELSE

$l = c_l$

END-FOR {*perform I iterations to obtain best value for* x_3}

It was mentioned before that the m x 3 DSA for the 3 sublot problem with attached setups runs in $O(m^3)$ time. Since the Bisection Algorithm calls m x 3 DSA $2I$ times, the whole solution is obtained in $O(Im^3)$ time. After I steps, the Bisection Algorithm will come within $2^{-I} + 2\varepsilon\ (1 - 2^{-I})$ of the optimal value of x_3.

The performance of the Bisection - m x 3 DSA was tested on a number of randomly generated sample problems. The results of this experimental study is provided in Table 4.1 below. Overall, 900 randomly generated problems were used in the study. The sample problems had machine numbers ranging from 10 to 20 and the unit processing times on these machines were uniformly distributed between 5 and 100 time units. Each problem was first solved by the Bisection - DSA and then by the LP solution package MPL for Windows for comparison. The error percentage obtained from the difference of the makespan values was used as the measure of performance in this analysis.

The first block in the table is related to the performance of the algorithm depending on the number of iterations used. The results show that as the number of iterations increases, the performance of the algorithm improves. Starting with 10 iterations in each run, the algorithm provided an excellent approximation to the optimal solution. However, the performance stays stable in the level of 0.01 % error even if the number of iterations is increased further.

The second block summarizes the results of the study done on the effect of the changes in the degree of significance ε. As seen in the table, when the degree of significance ε was increased from 0.01 to 0.001, the performance also increases 10 times.

Further increases in the degree of significance ε do not have much effect on the performance.

The performance of the algorithm on various size of problems is shown in the third block of Table 4.1. Even if the number of machines were increased from 10 to 20, the performance figures fell in the same vicinity. This shows that the algorithm can provide a robust performance on quite large problems.

Table 4.1 Performance evaluation of the Bisection - DSA

Number of problems	Number of Machines (m)	Iteration Number (I)	Deg. of Sig. (ε)	Setup/Proc. Ratio (SR)	Minimum % Error	Maximum % Error	Average % Error
50	10	5	0.001	U[0, 0.50]	4.69E-05	2.200829	0.376042
50	10	10	0.001	U[0, 0.50]	0.000215	0.053508	0.01214
50	10	15	0.001	U[0, 0.50]	0.000296	0.028868	0.01064
50	10	20	0.001	U[0, 0.50]	0.000308	0.026957	0.0104
				Iterations	4.69E-05	2.200829	0.102305
50	10	10	0.01	U[0, 0.50]	0.001805	0.284769	0.113009
50	10	10	0.001	U[0, 0.50]	0.000215	0.053508	0.01214
50	10	10	0.0001	U[0, 0.50]	8.55E-05	0.068556	0.011683
				Epsilon	8.55E-05	0.284769	0.045611
50	10	10	0.001	U[0, 0.50]	0.000215	0.053508	0.01214
50	15	10	0.001	U[0, 0.50]	0.000196	0.056294	0.012411
50	20	10	0.001	U[0, 0.50]	0.001123	0.071791	0.014424
				Machine #	0.000196	0.071791	0.012991
50	10	10	0.001	U[0, 0.10]	0.000245	0.062784	0.010067
50	10	10	0.001	U[0, 0.30]	9.21E-06	0.045338	0.009706
50	10	10	0.001	U[0, 0.50]	0.000215	0.053508	0.01214
50	10	10	0 001	U[0, 0.70]	6.73E-05	0.062139	0.012874
50	10	10	0.001	U[0, 0.90]	6.04E-05	0.042906	0.010103
50	10	10	0.001	U[0,1.10]	8.99E-05	0.059511	0.011168
50	10	10	0.001	U[0,1.30]	8.59E-05	0.047137	0.009766
50	10	10	0.001	U[0,1.50]	6.85E-05	0.048847	0.008115
				SR	9.21E-06	0.062784	0.010492
				OVERALL	9.21E-06	2.200829	0.037165

The last part of the study was comprised of the tests on the problems with varying magnitudes of setup time. The magnitude of the setup times in a problem is symbolized by the upper bound of the uniform distribution used to generate a ratio that will be

multiplied by the unit processing time on each machine to determine the setup time of that particular machine. The study included 400 sample problems with upper bounds on setup time to processing time ratio ranging from 0.10 to 1.50. The results in the table show that the algorithm performs well for any level of setup time to processing time ratio.

All in all, 900 sample problems with various configuration were tested and the average error percentage fell below 0.04 %. Since the lowest performance figures were obtained with 5 iterations and 0.01 degree of significance, eliminating these values gives the real performance of the algorithm with average error of 0.011 %.

4.1.1.3 General size problem with detached setups ($F, m \mid S, II, DS \mid n, C, R \mid C_{max}$)

The algorithm presented above is suitable to the three sublot problem with detached setups, however, it provides limited insight on the solution of problems with a larger number of sublots. In order to develop some techniques valuable for problems that allows more splitting, it is necessary to identify some properties of the problem. The LP formulation and network representation for n-sublot lot streaming problem with detached setups were already provided in Section 4.1.1 above. Definition 4.3 below states that a set of n or less machine indices can be used to define a path in the network representation.

Definition 4.3. *Let* $I[j]$, $j = 0, 1, \ldots, n\text{-}1$, *denote a machine index in the m x n lot streaming problem with detached setups. The set of machine indices I,*

$$I = \{I[0], I[1], \ldots, I[n\text{-}1]\}$$

where $I[j] \leq I[j+1]$, *for* $j = 0, 1, \ldots, n\text{-}2$, *defines the path, denoted by* $P(I)$, *that crosses over from column j to column* $j+1$ *in the network representation of the problem at node* $(I[j], j)$, *for* $j = 0, 1, \ldots, n\text{-}1$.

As seen in Definition 4.3, the paths in the network representation can be identified by the set of machine indices where the path crosses over from a column of nodes to the next. Since the makespan corresponds to the length of the longest path for the given set of sublot sizes, it is particularly important to focus on the longest paths. Therefore, we need a way to determine whether any given path between the nodes (1, 0) and (m, n) is the longest path. Theorem 4.1 below specifies the conditions under which a path can be the longest path in the network representation of a detached setup problem.

Theorem 4.1. *Each index* $I[j]$ *in the set* $I = \{I[0], I[1], \ldots, I[n-1]\}$, *where* $I[j] \le I[j+1]$ *for* $j = 0, \ldots, n\text{-}2$, *that defines a longest path must satisfy the condition*

$$\frac{T(i, I[j]-1)}{T(i+1, I[j])} \le z_j \le \frac{T(I[j], i'-1)}{T(I[j]+1, i')} \tag{4.47}$$

where $I[j\text{-}1] \le i \le I[j]\text{ -}1$ *with* $I[j] \ne 1$ *and* $I[j] + 1 \le i' \le I[j+1]$ *with* $I[j] \ne m$, *and where* $z_j = x_j / x_{j+1}$, *for* $j = 1, \ldots, n-1$. *Meanwhile, sublot size* 1 *must be also bounded as*

$$\sum_{h=I[1]+1}^{u} S'_h \Big/ T(I[1], u-1) \le x_1 \le \sum_{h=v+1}^{I[1]} S'_h \Big/ T(v, I[1]-1) \tag{4.48}$$

for all $u = I[0] + 1, \ldots, I[1]$ *and* $v = 1, \ldots, I[0]\text{-}1$.

Proof: Let $I = \{I[0], I[1], I[2], \ldots, I[n\text{-}1]\}$ be the set of machine indices defining a longest path in the network. In order that $P(I)$ might be the longest path in the network, the following inequality must be satisfied,

$$\sum_{k=1}^{I[0]} S'_k + \sum_{k=1}^{n} T(I[k-1], I[k]) x_k \ge \sum_{k=1}^{I[0]} S'_k + \sum_{k=1}^{j-1} T(I[k-1], I[k]) x_k + T(I[j-1], i) x_j \tag{4.49}$$

$$+ T(i, I[j]-1) x_{j+1} + \sum_{k=j+1}^{n} T(I[k-1], I[k]) x_k$$

for $j = 1,\ldots, n\text{-}1$, $I[j\text{-}1] \le i \le I[j]\text{-}1$ and $I[j] \ne 1$, where $I[n] = m$. Equation 4.49 reduces to the inequality below,

$$T(I[j-1], I[j])x_j \ge T(I[j-1], i)x_j + T(i, I[j]-1)x_{j+1}, \tag{4.50}$$

which is further simplified to

$$T(i+1, I[j])x_j \ge T(i, I[j]-1)x_{j+1}. \tag{4.51}$$

It finally leads to the following lower bound on the ratio of sublot sizes x_j and x_{j+1},

$$\frac{x_j}{x_{j+1}} \ge \frac{T(i, I[j]-1)}{T(i+1, I[j])} \tag{4.52}$$

for $j = 1, \ldots, n\text{-}1$, $I[j\text{-}1] \le i \le I[j]\text{-}1$ and $I[j] \ne 1$.

In order that $P(I)$ might be the longest path in the network, the following inequality must also be satisfied,

$$\sum_{k=1}^{I[0]} S'_k + \sum_{k=1}^{n} T(I[k-1], I[k])x_k \ge \sum_{k=1}^{I[0]} S'_k + \sum_{k=1}^{j} T(I[k-1], I[k])x_k + T(I[j]+1, i)x_j + T(i, I[j+1])x_{j+1} + \sum_{k=j+2}^{n} T(I[k-1], I[k])x_k \tag{4.53}$$

for $j = 1,\ldots, n\text{-}1$, $I[j] + 1 \le i \le I[j+1]$ and $I[j] \ne m$, where $I[n] = m$. Equation 4.53 reduces to the inequality below,

$$T(I[j], I[j+1])x_{j+1} \ge T(I[j]+1, i)x_j + T(i, I[j+1])x_{j+1}, \tag{4.54}$$

which is further simplified to

$$T(I[j], i-1)x_{j+1} \ge T(I[j]+1, i)x_j. \tag{4.55}$$

It finally leads to the following upper bound on the ratio of sublot sizes x_j and x_{j+1},

$$\frac{x_j}{x_{j+1}} \le \frac{T(I[j], i-1)}{T(I[j]+1, i)} \tag{4.56}$$

for $j = 1, \ldots, n\text{-}1$, $I[j] + 1 \le i \le I[j+1]$ and $I[j] \ne m$.

Using the definitions $z_j \equiv x_j / x_{j+1}$ for $j = 1,\ldots,n-1$, Equations 4.52 and 4.56 can be combined in the following inequality,

$$\frac{T(i, I[j]-1)}{T(i+1, I[j])} \le z_j \le \frac{T(I[j], i'-1)}{T(I[j]+1, i')} \tag{4.57}$$

for j = 1, …, n-1, $I[j-1] \le i \le I[j]$ -1, $I[j] \ne 1$ and $I[j] + 1 \le i' \le I[j+1]$, $I[j] \ne m$. Hence, Equation 4.47 must be satisfied for every index of the set I, in order for the path $P(I)$ to define a longest path in the network.

The following inequalities must also be satisfied, in order that $P(I)$ might be the longest path in the network,

$$\sum_{k=1}^{I[0]} S'_k + \sum_{k=1}^{n} T(I[k-1], I[k]) x_k \ge \sum_{k=1}^{u} S'_k + T(u, I[1]) x_1 + \sum_{k=2}^{n} T(I[k-1], I[k]) x_k, \tag{4.58}$$

for $u = I[0]+1$, …, $I[1]$, and

$$\sum_{k=1}^{I[0]} S'_k + \sum_{k=1}^{n} T(I[k-1], I[k]) x_k \ge \sum_{k=1}^{v} S'_k + T(v, I[1]) x_1 + \sum_{k=2}^{n} T(I[k-1], I[k]) x_k. \tag{4.59}$$

for v = 1, …, $I[0]$-1, where the right hand sides of Equations 4.58 - 4.59 represent the lengths of alternate paths in the network.

Equations 4.58 and 4.59 can be arranged to give the following bounds on sublot size 1,

$$x_1 \ge \sum_{h=I[0]+1}^{u} S'_h \Big/ T(I[0], u-1) \tag{4.60}$$

where $u = I[0]+1,\ldots, I[1]$, and

$$x_1 \le \sum_{h=v+1}^{I[0]} S'_h \Big/ T(v, I[0]-1) \tag{4.61}$$

where $v = 1, \ldots, I[0]$-1.

Finally, by combining Equations 4.60 and 4.61

$$\sum_{h=i_1+1}^{u} S'_h \Big/ T(i[0],u-1) \le x_1 \le \sum_{h=v+1}^{i_1} S'_h \Big/ T(v,i[0]-1) \qquad (4.62)$$

where $u = I[0]+1,\ldots, I[1]$ and $v = 1, \ldots, I[0]-1$.□

Theorem 4.1 provided bounds for $z_i = x_i/x_{i+1}$, for $i = 1,\ldots,n-1$, as well as for x_1 alone in order that the path defined by the set of machine indices I can stay as the longest in the network. The conditions in the theorem also imply that if any of the resulting lower bounds is larger than the respective upper bound, the path defined by the set of indices I can never be a longest path in the network regardless of the sublot sizes x_i, $i = 1, \ldots, n$.

Lemma 4.1 below provides a helpful modification for the problem with detached setups. In an alternative problem, which may also be called the reduced problem, the setup time on the first machine can be ignored in the process of finding an optimal makespan.

Lemma 4.1. *If M_1^* denotes the optimal makespan for an alternative lot streaming problem in which the setup times are replaced by $S_i^1 = \max\{S_i - S_1,\ 0\}$ for $i = 1, \ldots, m$, then optimal makespan for the original lot streaming problem with detached setups is $M^* = M_1^* + S_1$ and the optimal sublot sizes for the alternative problem are also optimal for the original problem.*

Proof: Since detached setups can be performed without waiting for the work pieces to arrive from the upstream machines, the setups on each machine can be shifted to start at time 0 assuming all machines are available at that time. Therefore, in the time interval $[0,S_1]$, every machine will be either idle or at the setup phase. The machines whose setup

times are less than or equal to S_1 will complete their setups in the interval [0, S_1] and then their setup times will be zero in the modified problem, i.e., $S_i^1 = 0$, $i = 1,\ldots,m$, if $S_i \le S_1$. However, if the setup on a machine i is longer than S_1, its setup will continue for $S_i - S_1$ time units beyond the interval [0, S_1] and therefore, its setup will be $S_i^1 = S_i - S_1$, $i = 1,\ldots,m$ in the modified problem. Combining the argument above, the definition of the setup times in the modified problem is $S_i^1 = \max\{S_i - S_1, 0\}$ for $i = 1,\ldots,m$.

Since no machine is active in the interval [0, S_1], the possibility of creating overlapped processing times and thus, shortening the makespan will start beyond time S_1. Essentially, only the remaining portions of the setups beyond time S_1 will be affecting the configuration of the sublot sizes, and therefore, the sublot sizes that provide the optimal makespan M_1^* to the reduced problem will also be optimal in the original problem. The optimal makespan of the original problem will then be $M^* = M_1^* + S_1$. □

At this point, another problem called the relaxed problem, in which setup times are deleted, can be introduced. The relaxed problem, which can be optimally solved by the techniques presented in Chapter 3, will also be useful during our efforts to solve the detached setup problem. Theorem 4.2 below states that the solution set of the lot streaming problem with detached setups is a subset of the solution set of the relaxed problem.

Theorem 4.2. *Let X_R be the feasible solution set for the relaxed problem. Also let X_S be the feasible solution set for the detached setup problem. Then, $X_S \subseteq X_R$.*

Proof: Consider an $\mathbf{x} \in X_S$. Naturally, since the formulation (P3.1) has the constraints of 3.1 and 3.3, which are equivalents of the constraints 4.6 and 4.8 of the formulation (P4.1), both sets are automatically satisfied by **x**. Now, consider the constraint 4.7 in (P4.1),

$$T_{11} \geq S_1 + p_1 x_1 > p_1 x_1,$$

therefore, x_1 also satisfies the constraint 3.2 in (P3.1). Similarly, by combining the constraints 4.9 and 4.10 in (P4.1),

$$T_{i1} \geq \max\left\{T_{i-1,1} + p_i x_1, S_i + p_i x_1\right\} \geq T_{i-1,1} + p_i x_1, \quad 2 \leq i \leq m,$$

we see that the part of the constraint 3.4 which is related to the first sublot is also satisfied. The remaining of the constraint 3.4 is also satisfied since it is the same as the constraint 4.11 in (P4.1). Therefore, any feasible solution of the detached setup problem is also a feasible solution to the relaxed problem. □

As a consequence of the theorem above, we have the following corollary which introduces a lower bound on the optimal makespan of the detached setup problem.

Corollary 4.1. $M_R^* \leq M_S^*$, *where* M_S^* *is the optimal makespan of the lot streaming problem with detached setups and* M_R^* *is the optimal makespan of the relaxed problem.*

Proof: From $X_S \subseteq X_R$.□

As stated in Lemma 4.2 below, the optimal solution to the relaxed problem occasionally coincides with the optimal solution to the reduced problem and consequently to the original problem. The condition to have the same optimal sublot sizes for both the relaxed and the original problem is that the first sublot should be large enough to allow the machines to complete their setups before the first sublot reaches to them. Therefore, the first sublot will flow through the shop without being delayed by the setups.

Lemma 4.2. *If the following holds for the sublot size vector* $\mathbf{x}^*$ *that optimally solves the relaxed problem,*

$$\max_{g=2,\ldots,m}\left\{\frac{\sum_{k=2}^{g} S_k'}{T(1,g-1)}\right\} < x_1^*, \tag{4.63}$$

then the sublot size vector $\mathbf{x}^*$ *also optimally solves the original problem.*

Proof: Let $\mathbf{x}^*$ be the vector of optimal sublot sizes in the relaxed problem. Then, the optimal makespan for the relaxed problem, M_R^*, which is the length of the longest path corresponding to the vector $\mathbf{x}^*$ in the network representation, is the following

$$M_R^* = \sum_{k=1}^{n} T(I[k-1], I[k])x_k^*,$$

where $T(i,j) = \sum_{k=i}^{j} p_k$ and $I[i]$ is the machine index at which the critical path of the network crosses over from column i to column $i+1$ while $I[0]=1$ and $I[n]=m$. If the following set of inequalities holds when the vector $\mathbf{x}^*$ is used in the network representation of the detached setup problem,

$$\begin{aligned} S_2' &< p_1 x_1^*, \\ S_2' + S_3' &< p_1 x_1 + p_2 x_1^*, \\ &\vdots \\ \sum_{k=2}^{m} S_k' &< T(1,m-1)x_1^*, \end{aligned}$$

or by combining the above inequalities

$$\max_{g=2,\ldots,m}\left\{\frac{\sum_{k=2}^{g} S'_k}{T(1,g-1)}\right\} < x_1^*,$$

that is, if the size of the first sublot in the optimum solution of the relaxed problem is large enough to allow every machine to finish its setup before the first sublot reaches to that machine, then, the path crossing over from column 0 to column 1 at a machine i, $i = 2,\ldots,m$, will never be a longest path in the network representation of the detached setup problem. Meanwhile, the critical path whose length provided the optimal makespan in the relaxed problem will still be the longest path for the detached setup problem and its length, M_S, for $S_1 = 0$, is the following

$$M_S = \sum_{k=1}^{n} T(I[k-1], I[k]) x_k^*,$$

which is equal to the optimal makespan of the relaxed problem, i.e.,

$$M_S = M_R^*.$$

Since the length of the longest path for the detached setup problem is yet to be minimized to obtain the optimal makespan, we have the following

$$M_S^* \le M_S = M_R^*.$$

However, Corollary 4.1 imposes the following lower bound,

$$M_R^* \le M_S^*.$$

Therefore, we must have

$$M_S^* = M_R^*,$$

which shows that the optimal solution to the relaxed problem is indeed the optimal solution to the detached setup problem. □

In the following, we present some observations on the critical paths corresponding to the optimal solution of the lot streaming problem with detached setups. These observations will form the foundations of a very powerful heuristic algorithm whose performance evaluation will also be provided.

Observation 4.1. *If the following holds in the detached setup problem,*

$$\sum_{k=2}^{h} S'_k \geq T(1,h-1),$$

where the machine index h is the index such that

$$\frac{\sum_{k=2}^{h} S'_k}{T(1,h-1)} = \max_{g=2,\ldots,m} \left\{ \frac{\sum_{k=2}^{g} S'_k}{T(1,g-1)} \right\}, \tag{4.64}$$

then the node $(h, 0)$ *will often be the only crossover node from column* 0 *to column* 1. *Meanwhile, the sublot size vector* $\mathbf{x}^*$ *that solves the relaxed problem formed with machines from h to m usually solves the original problem with detached setups.*

In the above case, the optimal makespan is $M^* = S_h + M^*_{hm}$, where M^*_{hm} is the optimal makespan for the relaxed subproblem containing machines h to m and S_h is the setup time on machine h. Hereinafter, the setup time on machine h will be referred to as "the most significant setup time".

The following observation provides one of the common cases where the optimal solution to the detached setup problem coincides with the optimal solution to the relaxed subproblem containing machines h to m, where machine h has the most significant setup time.

Observation 4.2. *If*

$$x_{r1}^{*} < \frac{\sum_{k=2}^{h} S_k'}{T(1,h-1)} < 1,$$

for the sublot size vector $\mathbf{x}_r^*$ *that optimally solves the relaxed problem, then, the machine h obtained through Equation* 4.64 *will possibly be the crossover index from column* 0 *to column* 1 *for any* x_1 *such that,*

$$\max_{l=h+1,\ldots,m}\left\{\sum_{k=h+1}^{l} S_k' \Big/ T(h,l-1)\right\} \le x_1 \le \min_{g=1,\ldots,h-1}\left\{\sum_{k=g+1}^{h} S_k' \Big/ T(g,h-1)\right\}. \tag{4.65}$$

Meanwhile, if the vector $\mathbf{x}_r^*$ *that optimally solves the relaxed subproblem with machines h to m satisfies Equation* 4.65, *then the vector* $\mathbf{x}_s^*$ *often solves the original problem.*

Lemma 4.3 below states the monotonic property of the lower and upper bounds provided in Equation 4.65, which are the necessary conditions under which a machine index can be a crossover index between columns 0 and 1. This monotonic property will help locate the crossover index between column 0 and column 1.

Lemma 4.3. *Bounds in Equation* 4.65 *are nonincreasing as the machine index moves from* 1 *to m.*

Proof: First the lower bounds will be considered. Suppose that we stand at machine 1 and look for the maximum among the ratios defined in Equation 4.65. Assume

$$\frac{\sum_{k=2}^{h} S_k'}{T(1,h-1)} = \max_{g=2,\ldots,m}\left\{\frac{\sum_{k=2}^{g} S_k'}{T(1,g-1)}\right\}.$$

Then, we must have the following set of inequalities,

$$\frac{\sum_{k=2}^{h} S'_k}{T(1,h-1)} \geq \frac{\sum_{k=2}^{h+1} S'_k}{T(1,h)} \tag{4.66}$$

$$\vdots$$

$$\frac{\sum_{k=2}^{h} S'_k}{T(1,h-1)} \geq \frac{\sum_{k=2}^{m} S'_k}{T(1,m-1)}. \tag{4.67}$$

From Equation 4.66, we have

$$\frac{\sum_{k=2}^{h} S'_k}{T(1,h-1)} \geq \frac{\sum_{k=2}^{h} S'_k + S'_{h+1}}{T(1,h-1)+T(h,h)}$$

and

$$\frac{\sum_{k=2}^{h} S'_k}{T(1,h-1)} \geq \frac{S'_{h+1}}{T(h,h)}. \tag{4.68}$$

From Equation 4.67, we have

$$\frac{\sum_{k=2}^{h} S'_k}{T(1,h-1)} \geq \frac{\sum_{k=2}^{h} S'_k + \sum_{k=h+1}^{m} S'_k}{T(1,h-1)+T(h,m-1)},$$

and

$$\frac{\sum_{k=2}^{h} S'_k}{T(1,h-1)} \geq \frac{\sum_{k=h+1}^{m} S'_k}{T(h,m-1)}. \tag{4.69}$$

Therefore, we get

$$\frac{\sum_{k=2}^{h} S'_k}{T(1,h-1)} \geq \max_{g=h+1,\ldots,m} \left\{ \frac{\sum_{k=h+1}^{g} S'_k}{T(h,g-1)} \right\}, \tag{4.70}$$

of which the right hand side is the definition of the lower bound at machine index h+1. This shows that the next lower bound will be less than or equal to the previous one.

The similar evaluation is applied for upper bounds as follows. Suppose that we stand at machine m and look for the minimum among the ratios defined in Equation 4.65. Assume

$$\frac{\sum_{k=t+1}^{m} S'_k}{T(t,m-1)} = \min_{g=1,\ldots,m-1} \left\{ \frac{\sum_{k=g+1}^{m} S'_k}{T(g,m-1)} \right\}.$$

Then, we must have the following set of inequalities,

$$\frac{\sum_{k=t+1}^{m} S'_k}{T(t,m-1)} \le \frac{\sum_{k=2}^{m} S'_k}{T(1,m-1)} \tag{4.71}$$

$$\vdots$$

$$\frac{\sum_{k=t+1}^{m} S'_k}{T(t,m-1)} \le \frac{\sum_{k=t}^{m} S'_k}{T(t-1,m-1)} \tag{4.72}$$

From Equation 4.71, we have

$$\frac{\sum_{k=t+1}^{m} S'_k}{T(t,m-1)} \le \frac{\sum_{k=2}^{t} S'_k + \sum_{k=t+1}^{m} S'_k}{T(1,t-1) + T(t,m-1)}$$

and

$$\frac{\sum_{k=t+1}^{m} S'_k}{T(t,m-1)} \le \frac{\sum_{k=2}^{t} S'_k}{T(1,t-1)}. \tag{4.73}$$

From Equation 4.72, we have

$$\frac{\sum_{k=t+1}^{m} S'_k}{T(t,m-1)} \leq \frac{S'_t + \sum_{k=t+1}^{m} S'_k}{T(t-1,t-1) + T(t,m-1)}$$

and

$$\frac{\sum_{k=t+1}^{m} S'_k}{T(t,m-1)} \leq \frac{S'_t}{T(t-1,t-1)} \tag{4.74}$$

Therefore, we get

$$\frac{\sum_{k=t+1}^{m} S'_k}{T(t,m-1)} \leq \min_{g=1,\ldots,t-1} \left\{ \frac{\sum_{k=g+1}^{t} S'_k}{T(g,t-1)} \right\}, \tag{4.75}$$

of which the right hand side is the definition of the upper bound at machine index t. This shows that the next upper bound will be less than or equal to the previous one. □

Another important observation is presented below. The following case is observed when the solution obtained from the relaxed subproblem does not yield a size for the first sublot that falls in the bounds defined by Equation 4.65.

Observation 4.3. *If*

$$x^*_{r1} < \frac{\sum_{k=2}^{h} S'_k}{T(1,h-1)} < 1,$$

for the sublot size vector $\mathbf{x}^*_r$ *that optimally solves the relaxed problem, then, the machine* h *obtained through Equation* 4.64 *will rarely be the crossover index from column* 0 *to column* 1 *for any* x_1 *such that,*

$$0 < x_1 < \max_{l=h+1,\ldots,m} \left\{ \sum_{k=h+1}^{l} S'_k \Big/ T(h,l-1) \right\} \tag{4.76}$$

or

$$\min_{g=1,\ldots,h-1}\left\{\sum_{k=g+1}^{h} S_k' \Big/ T(g,h-1)\right\} < x_1 < 1. \tag{4.77}$$

The observation above provides the information that the solution to the relaxed subproblem may not always provide an obvious solution to the original problem since the machine h is not a crossover index with the sublot sizes obtained from the subproblem. Meanwhile, the following observation emphasizes that a shorter makespan may be found for the original problem by manipulating the solution to the relaxed subproblem when the case explained in Observation 4.3 is in effect.

Observation 4.4. *If either of Equations* 4.76 and 4.77 *holds for the vector* $\mathbf{x}_s^*$ *that optimally solves the relaxed subproblem with machines h to m, a vector* $\mathbf{x}^*$ *can be obtained in the following way which often provides a shorter makespan for the original problem.*

1.a. *If* $0 < x_{s1}^* < \max_{l=h+1,\ldots,m}\left\{\sum_{k=h+1}^{l} S_k' \Big/ T(h,l-1)\right\}$, *set* $x_1^* = \max_{l=h+1,\ldots,m}\left\{\sum_{k=h+1}^{l} S_k' \Big/ T(h,l-1)\right\}$.

1.b. *If* $\min_{g=1,\ldots,h-1}\left\{\sum_{k=g+1}^{h} S_k' \Big/ T(g,h-1)\right\} < x_{s1}^* < 1$, *set* $x_1^* = \min_{g=1,\ldots,h-1}\left\{\sum_{k=g+1}^{h} S_k' \Big/ T(g,h-1)\right\}$.

2. *Obtain the remaining* $x_j^*, j = 2,\ldots,n$, *from the following system of linear equations assuming that some of the critical boxes in the optimal solution of the relaxed subproblem is still usable in the original problem,*

$$x_j^* = x_{j-1}^* \frac{x_{s,j}^*}{x_{s,j-1}^*}, \quad j = 2,\ldots,n, \tag{4.78}$$

$$x_2^* + x_3^* + \ldots + x_n^* = 1 - x_1^*. \tag{4.79}$$

In the above system, there are n equations while there are n-1 variables. Therefore, one of the equalities in Equation 4.78 should be left out of the evaluation to obtain for a proper set of sublot sizes for the original problem. Since there are n-1 different ways of choosing n-2 equations out of n-1 equations, there will be n-1 different sets of sublot sizes that need to be compared in terms of the makespan value they provide. The set of sublot sizes that yields the smallest makespan is chosen as the solution set for the original problem.

The observable properties of the critical paths in the network representation of the lot streaming problem with detached setups helped us design an algorithmic approach to the problem. The following has been a powerful heuristic algorithm for the general size lot streaming problem with detached setups, which is currently solvable to optimality only by using an LP solver package.

Significant Setup Time Algorithm

1. Solve the relaxed problem and obtain the optimal sublot size vector $\mathbf{x}_r^*$ and the makespan M_r^*.

2. Determine the machine index h with most significant setup time such as

$$\text{MSSR} = \sum_{k=2}^{h} S_k' \Big/ T(1,h-1) = \max_{l=2,\ldots,m} \left\{ \sum_{k=2}^{l} S_k' \Big/ T(1,l-1) \right\}.$$

3. IF $x_{r1}^* >$ MSSR,

   Set $\mathbf{x}^* = \mathbf{x}_r^*$ and the corresponding optimal makespan is $M^* = M_r^* + S_1$.

   Go to Step 7.

   IF $x_{r1}^* \leq$ MSSR,

Form a subproblem with machines from h to m.

Solve the subproblem as a relaxed problem to obtain the optimal sublot size vector $\mathbf{x}_s^*$.

4. Obtain the lower bound index j, $h < j \le m$, such as

$$\mathrm{LB} = \sum_{k=h+1}^{j} S_k' \Big/ T(h, j-1) = \max_{l=h+1,\ldots m} \left\{ \sum_{k=h+1}^{l} S_k' \Big/ T(h, l-1) \right\},$$

and the upper bound from g, $1 \le g < h$, such as

$$\mathrm{UB} = \sum_{k=g+1}^{h} S_k' \Big/ T(g, h-1) = \min_{l=1,\ldots,h-1} \left\{ \sum_{k=l+1}^{h} S_k' \Big/ T(l, h-1) \right\}.$$

5. If $\mathrm{LB} \le x_{s1}^* \le \mathrm{UB}$,

Set $\mathbf{x}^* = \mathbf{x}_s^*$ and obtain the corresponding makespan M^* by using a longest path algorithm.

Go to Step 7.

6. If $x_{s1}^* < \mathrm{LB}$,

Set $x_1^* = \mathrm{LB}$.

If $x_{s1}^* > \mathrm{UB}$,

Set $x_1^* = \mathrm{UB}$.

Set $x_{a1}^* = x_1^*$ and calculate x_{ai}^*, i = 2, …, n, by using the sublot size ratios z_i, $i = 2,\ldots,n-1$.

Compute M_a^* by using $\mathbf{x}_a^*$ in a longest path algorithm.

Set $x_{b1}^* = x_1^*$ and calculate x_{bi}^*, i = 2, …, n, by using the sublot size ratios z_i, $i = 1,\ldots,n-2$.

Compute M_b^* by using $\mathbf{x}_b^*$ in a longest path algorithm.

IF $M_a^* < M_b^*$,

Let $x_i^* = x_{ai}^*$, $i = 1,\ldots,n$, and $M^* = M_a^*$,

ELSE,

Let $x_i^* = x_{bi}^*$, $i = 1,\ldots,n$, and $M^* = M_b^*$.

7. Take the sublot size vector $\mathbf{x}^*$ providing the makespan value M^* as the solution of the detached setup problem.

Significant Setup Time Algorithm has the complexity of $O(n^2(m+n))$, since it needs solving the m x n relaxed problem which can be performed by the Block Algorithm presented in Section 3.6.3. The algorithm above was tested on 100 randomly generated problems. The sample problems all had 20 machines and 7 sublots. The processing time data was generated using a uniform distribution with a minimum of 5 and a maximum of 100 time units. The ratio of detached setup time to processing time on each machine is uniformly distributed between 0 and 2. Significant Setup Time Algorithm obtained optimal solutions to 93 sample problems. Among the problems which the algorithm failed to provide an optimal solution, the error percentage ranged from a minimum of 0.57 % to a maximum of 1.75 %. The overall percentage was found as 0.07 %. It can be seen that the main reason for not being able to optimally solve every sample problem in the set is that only two combinations out of n-1 possibilities were used when obtaining the remaining sublot sizes after the size of the first sublot has been fixed by the bounds. The combinations of the sublot size ratios used in the algorithm were chosen based on their higher frequency of being active in the formation of the optimal sublot sizes.

4.1.2 Lot Streaming Problems with Attached Setups

If the setups are all attached to the first sublot, the necessary adaptation on the formulation (P3.1) will be the replacement of (3.2) and (3.4) by the following,

$$T_{11} \geq p_1 x_1 + S_1 \tag{4.80}$$

and

$$T_{i1} \geq T_{i-1,1} + p_i x_1 + S_i, \qquad 2 \leq i \leq m, \tag{4.81}$$

$$T_{ij} \geq T_{i-1,j} + p_i x_j, \qquad 2 \leq i \leq m,\ 2 \leq j \leq n, \tag{4.82}$$

where S_i is the length of the setup time that must be performed on machine i before processing the first sublot.

Even though, the adaptation for the attached setup case is easier, there is no obvious solution unlike the detached setup case when the setups are not binding. The only judgmental idea is that when the setups are long, the first sublot should be reduced by a considerable amount in order to finish the setups as early as possible. When there are attached setups in the problem, the makespan will always be somewhat longer than the one obtained without setups.

Chen and Steiner studied both cases in three-machine flow shop environments [Che94, Che95]. Using the network fundamentals developed by Glass et. al. [Gla94], they analyzed the structural changes when setups are included. Later they determined the solution to the problem by checking all possible critical paths. Their search results in $O(n)$ and $O(\log(n))$ complexity in attached setup case and detached setup case, respectively. Since these studies use many properties coherent to the three-machine problem, they do not help obtain viable solutions to arbitrary machine number case.

4.1.2.1 Two-sublot problem with attached setups ($F, m \mid S, II, AS \mid 2, C, R \mid C_{max}$)

In the previous section, it was shown how setups change the LP formulation. Starting with this section, we will focus on how to solve varying sizes of m machine problems with attached setups. First, the structure of the problem allowing two consistent sublots will be studied. The formulation for the two sublot problem is as follows,

(P4.5) minimize T_{m2}

subject to

$$x_1 + x_2 = 1 \tag{4.83}$$

$$T_{11} \geq p_1 x_1 + S_1 \tag{4.84}$$

$$T_{i2} \geq T_{i,1} + p_i x_2, \qquad 1 \leq i \leq m, \tag{4.85}$$

$$T_{i1} \geq T_{i-1,1} + p_i x_1 + S_i, \qquad 2 \leq i \leq m, \tag{4.86}$$

$$T_{i2} \geq T_{i-1,2} + p_i x_2, \qquad 2 \leq i \leq m, \tag{4.87}$$

$$T_{ij} \geq 0, \qquad 1 \leq i \leq m,\ j = 1,2, \tag{4.88}$$

$$x_j \geq 0, \qquad j = 1,2. \tag{4.89}$$

The above formulation can also be represented by a project network considering the constraint sets (4.84), (4.86) and (4.87) as the vertical precedence relationships and the constraint set (4.85) as the horizontal precedence relationships. The network representation is shown in Figure 4.4 where the nodes (i, 1) represent the completion of the setup of machine i and the processing of the first sublot, and the nodes (i, 2) represent the processing of the second sublot on machine i. In this network representation, the length of the longest path corresponds to the makespan, however since sublot sizes are decision variables, this network has variable-length activity times.

Definition 4.4. *In the network representation of the two sublot problem with attached setups, a path emanating from node* (1, 1) *and reaching to node* (m, 2) *can be represented by the machine index,* i, $1 \le i \le m$, *at which the path crosses over from column* 1 *to column* 2.

As seen in Definition 4.4, the length of any possible path between node (1, 1) and node (m, 2) can be represented by $f_i(x_1, x_2)$, where i is the machine index where the path switches from the first column to the second column in the network. The length of the path which crosses over at machine index i is defined as,

$$f_i(x_1, x_2) = \sum_{k=1}^{i} S_k + T(1,i)x_1 + T(i,m)x_2, \tag{4.90}$$

where $T(i,j) = \sum_{k=i}^{j} p_k$.

Since the sum of the two sublot sizes must be equal to the size of the production lot, $x_1 + x_2 = 1$, x_2 can be replaced by $1 - x_1$, and in the formulations simply x can be used instead of x_1. Then, the length of the path becomes a linear function of x as follows,

$$f_i(x) = \sum_{k=1}^{i} S_k + T(1,i)x + T(i,m)(1-x),$$

or by grouping the terms,

$$f_i(x) = \big(T(1,i-1) - T(i+1,m)\big)x + \sum_{k=1}^{i} S_k + T(i,m),$$

or by defining $c_i = T(1,i-1) - T(i+1,i)$,

$$f_i(x) = c_i x + \sum_{k=1}^{i} S_k + T(i,m). \tag{4.91}$$

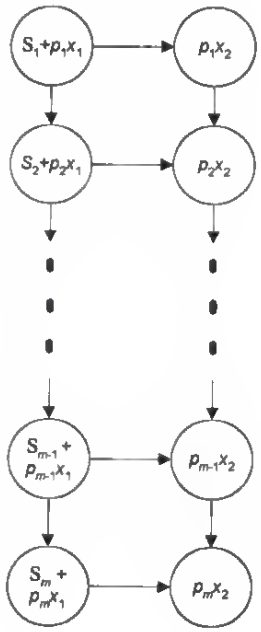

Figure 4.4. Network representation of the two sublot problem with attached setups

Therefore, for the specified sublot combination the length of the longest path is defined as

$$f_M(x) = \max_{i=1,\ldots,m} f_i(x) \tag{4.92}$$

which is defined in the domain $X = \{x \in \mathcal{R} \mid 0 \leq x \leq 1\}$. In order to find the optimal makespan, length of the longest path should be minimized over the feasible domain for the first sublot, that is

$$M^* = \min_{0 \leq x \leq 1} f_M(x). \tag{4.93}$$

Finally, combining both stages, the whole solution process is represented by

$$M^* = \min_{0 \leq x \leq 1} \max_{i=1,\ldots,m} f_i(x). \tag{4.94}$$

The above min-max problem can also be represented in the form of a linear program with two decision variables and m constraints as follows,

(P4.6) minimize M

subject to $$M - c_i x \geq \sum_{k=1}^{i} S_k + T(i,m), \qquad i = 1,\ldots,m \tag{4.95}$$

$$M \geq 0 \tag{4.96}$$

$$0 \leq x \leq 1 \tag{4.97}$$

where x is the size of the first sublot and M is the makespan of the problem.

As seen above, the two sublot problem with attached setups reduces to the minimization problem over a convex envelope created by the maximum of the linear functions associated with each machine index, as shown in Figure 3.3. At this point, Williams and Tüfekçi's [Wil95] solution technique for m machine 2 sublot problem that was introduced in Section 3.3.3 is directly applicable with a linear-time complexity. In this case, the only difference is an increase of $\sum_{k=1}^{i} S_k$ in the function corresponding to machine index i.

4.1.2.2 Three-sublot problem with attached setups ($F, m \mid S, II, AS \mid 3, C, R \mid C_{max}$)

In order to begin the study of the m machine 3 sublot lot streaming problem with attached setups, it is helpful to overview its LP formulation provided below.

(P4.7) minimize T_{m3}

subject to $$x_1 + x_2 + x_3 = 1 \tag{4.98}$$

$$T_{11} \geq p_1 x_1 + S_1 \tag{4.99}$$

$$T_{ij} \geq T_{i,j-1} + p_i x_j, \qquad 1 \leq i \leq m, \;\; 2 \leq j \leq 3 \tag{4.100}$$

$$T_{i1} \geq T_{i-1,1} + p_i x_1 + S_i, \qquad 2 \leq i \leq m, \tag{4.101}$$

$$T_{ij} \geq T_{i-1,j} + p_i x_j, \qquad 2 \leq i \leq m, \ 2 \leq j \leq 3 \tag{4.102}$$

$$T_{ij} \geq 0, \qquad 1 \leq i \leq m, \ 1 \leq j \leq 3, \tag{4.103}$$

$$x_j \geq 0, \qquad 1 \leq j \leq 3. \tag{4.104}$$

The project network representation corresponding to the m machine 3 sublot lot streaming problem with attached setups is presented in Figure 4.5 below.

As in the case of two sublots, here again the length of the longest path in the network representation determines the makespan for the specified combination of the sublot sizes. In order to obtain the minimum makespan, it is necessary to look for the sublot sizes that yield the minimum longest path in the network.

Definition 4.4. *In the network representation of the three sublot problem with attached setups, a path emanating from node* (1, 1) *and reaching to node* (m, 3) *can be represented by the machine indices, i, j, $1 \leq i \leq j \leq m$, at which the path crosses over from column* 1 *to column* 2 *and from column* 2 *to column* 3, *respectively.*

Using Definition 4.4, the length of the path traversing from the node (1, 1) to node (m, 3) with crossover indices i and j can be formulated by the following equation,

$$M_{ij}(x_1, x_2, x_3) = T(1,i)x_1 + \sum_{k=1}^{i} S_k + T(i,j)x_2 + T(j,m)x_3, \tag{4.105}$$

where $T(i,j) = \sum_{k=i}^{j} p_k$, while p_k is the unit processing time on machine k. In Equation 4.105 above, i and j, $1 \leq i \leq j \leq m$, represent the machine indices that the critical path switches from the first column to the second, and from the second column to the third, respectively.

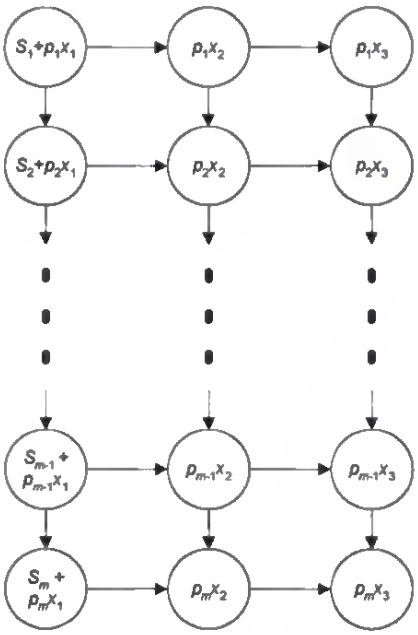

Figure 4.5. Network representation of the three sublot problem with attached setups

When the size of the third sublot, x_3, is kept as constant, the length of the path will reduce to the following one-variable function due to the constraint (4.98),

$$M_{ij}(x_1,x_2,x_3) = T(1,i)x_1 + \sum_{k=1}^{i} S_k + T(i,j)(1-c-x_1) + T(j,m)c, \tag{4.106}$$

or

$$M_{ij}(x_1) = \big(T(1,i) - T(i,j)\big)x_1 + \sum_{k=1}^{i} S_k + T(i,j)(1-c) + T(j,m)c, \tag{4.107}$$

where x_3 is replaced by the constant c, and x_2 becomes $(1 - c - x_1)$.

The above transformation yields $\big(m(m+1)/2\big)$ equations in terms of x_1, given x_3 is known. Since the makespan corresponds to the longest path in the network representation, the maximum of all resulting functions is needed,

$$M(x_1) = \max_{1 \le i \le j \le m} M_{ij}(x_1). \tag{4.108}$$

Finally, the optimal makespan for $x_3 = c$ is obtained by minimization over the domain $0 \le x_1 \le 1 - c$, where all the functions are defined,

$$M^*(c) = \min_{0 \le x_1 \le 1-c} \max_{1 \le i \le j \le m} M_{ij}(x_1). \tag{4.109}$$

Therefore, the solution process can also be expressed by the following LP formulation with 2 variables and $O(m^2)$ constraints,

(P4.8) minimize $M(c)$

subject to

$$M(c) - \big(T(1,i) - T(i,j)\big)x_1 \ge \sum_{k=1}^{i} S_k + T(i,j)(1-c) + T(j,m)c, \quad 1 \le i \le j \le m, \tag{4.110}$$

$$M(c) \ge 0 \tag{4.111}$$

$$0 \le x_1 \le 1 - c \tag{4.112}$$

It is clear that the above linear program can be solved by the search algorithm developed by Williams and Tüfekçi [Wil95] in $O(m^2)$ for a given c. In order to obtain a solution close to the global optimal, the value of c must also be analyzed. Therefore, there is a need to restructure the above mentioned solution method specifically for the 3 sublot attached setup problem. The modified algorithm will be called as m x 3 Attached Setup Algorithm (m x 3 ASA).

m x 3 Attached Setup Algorithm (for fixed x_3 value)

0. Let $L = \{l_1, l_2, \ldots, l_{m(m+1)/2}\}$ be the set of the functions defining the length of paths in the network representing the lot streaming problem with attached setups and $0 < c < 1$, where c is the parametric value of x_3, be fixed.

Also let $\delta = 1/Q$, where Q is the number of items in the production lot.

Set $x_l = 0$, the lower limit for the first sublot size.

Set $x_u = 1 - c$, the upper limit for the first sublot size.

1. For each $t \in N = \left\{1,3,\ldots,2\left\lfloor \frac{NUM(L)}{2} \right\rfloor - 1\right\}$, set

$$x_{t,t+1} = \frac{R_{t+1} - R_t}{e_t - e_{t+1}}$$

where $R_t = \sum_{k=1}^{i_t} S_k + T(i_t, j_t)(1-c) + T(j_t, m)c$,

and $e_t = \left(T(1,i_t) - T(i_t, j_t)\right)$, $i_t = 1,\ldots,m$, $i_t \le j_t$, for i_t, j_t crossover indices associated with the path represented by function l_t.

2. For each $t \in N$,

IF $x_{t,t+1} \ge x_H$,

Update N and L as $N\backslash\{t\}$ and $L\backslash\{k\}$, respectively, where

$$k = \begin{cases} l_t & \text{if } e_t > e_{t+1} \\ l_{t+1} & \text{otherwise.} \end{cases}$$

IF $x_{t,t+1} \le x_L$,

Update N and L as $N\backslash\{t\}$ and $L\backslash\{k\}$, respectively, where

$$k = \begin{cases} l_t & \text{if } e_t < e_{t+1} \\ l_{t+1} & \text{otherwise.} \end{cases}$$

3. If $NUM(L) = 2$,

Algorithm is complete, go to Step 7.

IF $N = \{\varnothing\}$,

Relabel $L = \{l_1, l_2, \ldots, l_{NUM(L)}\}$ and go to Step 1.

ELSE

Set

$$\bar{x} = \sum_{t \in T} \frac{x_{t,t+1}}{NUM(T)}$$

4. Let $E = \{r \in L \mid e_r \bar{x} + R_r \geq e_l \bar{x} + R_l, \forall l \in L\}$.

Set $slope1 = \min_{r \in E}\{e_r\}$, $slope1i = r$, and $slope2 = \max_{r \in E}\{e_r\}$, $slope2i = r$.

5. IF $(slope1)(slope2) \leq 0$,

Set $x^* = \bar{x}$

Set $M^*(c) = \max_{l \in L}\{e_l \bar{x} + R_l\}$, stop.

6. IF $slope1 > 0$,

Set $x_u = \bar{x}$ and for each $t \in T$ such that $x_{t,t+1} \geq \bar{x}$.

Update L as $L \backslash \{k\}$ where

$$k = \begin{cases} l_t & \text{if } e_t > e_{t+1} \\ l_{t+1} & \text{otherwise.} \end{cases}$$

IF $slope1 < 0$,

Set $x_L = \bar{x}$ and for each $t \in T$ such that $x_{t,t+1} \leq \bar{x}$.

Update L as $T \backslash \{k\}$ where

$$k = \begin{cases} l_t & \text{if } e_t < e_{t+1} \\ l_{t+1} & \text{otherwise.} \end{cases}$$

7. Relabel $L = \{l_1, l_2, \ldots, l_{NUM(L)}\}$.

IF $NUM(L) = 2$,

Optimal size of sublot 1 is

$$x^* = \frac{R_2 - R_1}{e_1 - e_2}.$$

Set $slope1 = \min_{r=1,2}\{e_r\}$, $slope1i = r$, and $slope2 = \max_{r=1,2}\{e_r\}$, $slope2i = r$.

ELSE

Go to Step 1.

8a. The optimal makespan is $M^*(c) = e_1 x^* + R_1$.

8b. *Adjustment step*

IF $(x^* < x_l)$ OR $(x^* < \delta)$

IF $(\delta > x_l)$

$x^* = \delta$

ELSE

$x^* = x_l$

$M^*(c) = e_{slope2i} x^* + R_{slope2i}$.

IF $(x^* > x_u)$ AND $(x_u > \delta)$

$x^* = x_u$

$M^*(c) = e_{slope1i} x^* + R_{slope1i}$.

IF $(e_{slope1i} > 0)$ AND $(e_{slope2i} > 0)$

IF $(\delta > x_l)$

$x^* = \delta$

ELSE

$x^* = x_l$

$M^*(c) = e_{slope1i}x^* + R_{slope1i}$.

IF ($e_{slope1i} < 0$) AND ($e_{slope2i} < 0$)

IF ($\delta > x_u$)

$x^* = \delta$

ELSE

$x^* = x_u$

$M^*(c) = e_{slope2i}x^* + R_{slope2i}$.

The solution obtained from m x 3 Attached Setup Algorithm (ASA) is optimal only if the specified x_3 value, c, is in fact optimal. In order to be close enough to a globally optimal solution, this algorithm should be employed in the following bisection framework.

Bisection Algorithm

INPUT $m, p_1, \ldots, p_m, K, \varepsilon$.

LET $l = 0$ and $u = 1$.

FOR $i = 1$ TO K

BEGIN {*perform K iterations to obtain best value for* x_3}

$c_m = (l + u) / 2$

$c_l = c_m - \varepsilon$

$c_r = c_m + \varepsilon$

$M_l = m$ x 3 ASA ($m, p_1, \ldots, p_m, S_1, \ldots, S_m, c_l$)

$M_r = m$ x 3 ASA ($m, p_1, \ldots, p_m, S_1, \ldots, S_m, c_r$)

IF $M_l < M_r$

$u = c_r$

ELSE

$l = c_l$

END-FOR {*perform K iterations to obtain best value for* x_3}

It was mentioned before that the m x 3 ASA for the 3 sublot problem with attached setups is $O(m^2)$. Since the Bisection Algorithm calls m x 3 ASA $2K$ times, the whole solution is obtained in $O(Km^2)$ time. After K steps, the Bisection Algorithm will come within $2^{-K} + 2\varepsilon\ (1-2^{-K})$ of the optimal value of x_3.

The performance of the algorithm presented above was tested for several different scenarios. The results of the experiments performed for m x 3 Attached Setup Algorithm are presented on Table 4.2.

In the experiments with the 3-sublot heuristic, the accuracy of the algorithm was first tested with varying number of iterations. For this purpose, 600 sample problems were randomly generated with 10 machines whose processing times are uniformly distributed between 5 and 100 time units while the ratio of setup time to processing time is uniformly distributed between 0 and 1 for each machine. These problems were divided into 4 groups and tested each group under a given number of iterations. We used 5, 10, 15 and 20 iterations solving grouped problems. The epsilon value used in bisection step was 0.001 for all problems in this category. The analysis of results shows that the heuristic becomes more accurate when the number of iterations are increased. The overall performance in this category has been very satisfactory which yielded an average error of 0.061 %. When we disregard the results obtained via 5 iterations, the performance turns out to be 10 times better. Therefore, it is very beneficial to employ at least 10 iterations in the applications of the heuristic.

Table 4.2 Performance evaluation of the 3-sublot Bisection - ASA

Pr. Size m	Range of p_i	Range of S/P Ratio	Number of iterations	Epsilon value	# of probs. solved	Min. % Error	Max. % Error	Average % Error
10	[5, 100]	[0, 1]	K=5	0.001	150	0	1.589871	0.2278417
10	[5, 100]	[0, 1]	K=10	0.001	150	0	0.04121	0.0061008
10	[5, 100]	[0, 1]	K=15	0.001	150	0	0.022357	0.0059245
10	[5, 100]	[0, 1]	K=20	0.001	150	0	0.022211	0.0058988
				Iterations	600	0	1.589871	0.0614415
15	[5,100]	[0, 1]	K=15	0.01	150	0	0.247705	0.0609402
15	[5,100]	[0, 1]	K=15	0.001	150	0	0.04999	0.0068883
15	[5,100]	[0, 1]	K=15	0.0001	150	0	0.003494	0.0006287
				Epsilon	450	0	0.247705	0.022819
10	[5,100]	[0, 1]	15	0.001	25	0	0.019194	0.0049359
15	[5,100]	[0, 1]	15	0.001	25	0	0.01828	0.0054072
20	[5,100]	[0, 1]	15	0.001	25	0	0.021132	0.0064385
				Machine #	75	0	0.021132	0.0055939
			OVERALL		1125	0	1.589871	0.0327773

This heuristic was also tested in terms of the effects of the epsilon value used in the bisection part of the heuristic algorithm. In this section, 150 sample problems were randomly generated for each group that will be tested under the epsilon values of 0.01, 0.001 and 0.0001, respectively. The problems all have 15 machines, processing times uniformly distributed between 5 and 100 time units and setup time to processing time ratios uniformly distributed between 0 and 1. Then, all problems in this category were solved using 15 iterations. The analysis of the results obtained by the solution of these 450 sample problems reveals that when the epsilon value is decreased an order of magnitude, the overall performance increases an order of magnitude.

The heuristic algorithm designed for the 3-sublot attached setup problem was finally tested to observe the effect of the increasing machine numbers. In this category, 25 problems were randomly generated for each group of 10, 15 and 20 machine numbers and the problems were solved with 15 iterations of the m x 3 ASA. The analysis of the results obtained shows that the heuristic proves to be valuable on problems with even larger

machine numbers. Overall, this heuristic had a very good performance on the total of 1125 sample problems. The results are presented in Table 4.2.

4.1.2.3 General size problem with attached setups ($F, m \mid S, II, AS \mid n, C, R \mid C_{max}$)

The lot streaming problem having more than three sublots can also be solved by a slightly modified implementation of the Attached Setup Algorithm. In order to apply this solution technique, the problem must be reduced to a form where there is only one sublot size that is to be determined by the search algorithm. This reduction is performed by bisection loops nested inside each other. For example, when the sublot number is 4, two bisection loops nested inside each other are necessary in order to assign feasible constant values to two of the sublot sizes, while the third one is being determined from the production lot size, after the sublot size chosen as the decision variable has been found by the search algorithm. However, it is expected that the number of functions defining the longest paths in the network will grow exponentially for n sublot problem. Therefore, a simpler heuristic may also serve the purpose of obtaining a reasonable solution to the n-sublot lot streaming problem with attached setups. During our computational studies, we had some observations that led to a valuable heuristic. In these studies, we compared the ratios of sublot sizes obtained through the 3-sublot and n-sublot ($n > 3$) configurations of the same problem data. We observed that the ratios $z_2 = x_2/x_3, \ldots, z_{n-1} = x_{n-1}/x_n$ obtained from the n-sublot configuration are frequently equal to the ratio $z_2' = x_2'/x_3'$ obtained from the 3-sublot configuration of the same problem data. However, the ratios of the first two sublots were quite different in the 3-sublot and n-sublot cases. When the sublot number increases, the ratio of the first two sublot sizes has a tendency to decrease gradually. Therefore, for any sublot number n, the ratio of the first two sublot sizes will

be bounded above by the ratio obtained from the 3-sublot solution, that is $z_1 = x_1/x_2 \leq z_1' = x_1'/x_2'$. Meanwhile, since the first sublot itself has a lower bound, δ, the ratio of the first two sublot sizes will also be bounded below by δ, that is $z_1 > \delta$. Using this observation, a heuristic algorithm was developed for the n-sublot attached setup problem. This algorithm first solves the problem with 3 sublots and then tries to obtain the best value for the ratio of the first two sublot sizes in the n-sublot problem while assuming the values of the remaining ratios are equal to the ratio of the second and third sublot sizes in the 3-sublot problem. This new heuristic can be summarized as follows.

m x n Attached Setup Algorithm

0. Solve m x 3 problem with the m x 3 ASA to obtain x_1', x_2' and x_3'.

   Set $z_1' = x_1'/x_2'$ and $z_2' = x_2'/x_3'$.

   Define $z_i = x_i/x_{i+1}$, for $1 \leq i \leq n-1$ and $\varepsilon < 0$ and small enough in the n-sublot problem.

   Let $\delta = 1/Q$, where Q is the number of items in the production lot.

1. FOR $i = 2$ TO n-1

   $z_i = z_2'$.

   Let $l = \delta$ and $u = z_1'$.

2. FOR $i = 1$ TO K

   BEGIN {*perform K iterations to obtain the best* z_1 *value*}

   $c_m = (l + u) / 2$

   $c_l = c_m - \varepsilon$

   $z_1 = c_l$

Obtain sublot sizes, $x_{l,i}$, $1 \le i \le n$, from the following equations,

$$x_{l,i} = \frac{\prod_{j=i}^{n-1} z_j}{\sum_{k=1}^{n-1} \prod_{j=k}^{n-1} z_j + 1}, \quad 1 \le i \le n-1,$$

and

$$x_{l,n} = \frac{1}{\sum_{k=1}^{n-1} \prod_{j=k}^{n-1} z_j + 1}.$$

IF $x_{l,1} < \delta$,

BEGIN {*lower bound violation, adjust* $x_{l,1}$ *and recalculate other sublot sizes*}

Set $x_{l,1} = \delta$ and obtain other sublot sizes from the following equations,

$$x_{l,i} = \frac{(1-\delta)\prod_{j=i}^{n-1} z_j}{\sum_{k=1}^{n-1} \prod_{j=k}^{n-1} z_j + 1}, \quad 2 \le i \le n-1,$$

and

$$x_{l,n} = \frac{1-\delta}{\sum_{k=1}^{n-1} \prod_{j=k}^{n-1} z_j + 1}.$$

END-IF {*lower bound violation, adjust* $x_{l,1}$ *and recalculate other sublot sizes*}

Find M_l, the length of the longest path, by using $\mathbf{x}_l$.

$c_r = c_m + \varepsilon$

$z_1 = c_r$

Obtain sublot sizes, $x_{r,i}$, $1 \le i \le n$, from the following equations,

$$x_{r,i} = \frac{\prod_{j=i}^{n-1} z_j}{\sum_{k=1}^{n-1}\prod_{j=k}^{n-1} z_j + 1}, \quad 1 \le i \le n-1,$$

and

$$x_{r,n} = \frac{1}{\sum_{k=1}^{n-1}\prod_{j=k}^{n-1} z_j + 1}.$$

IF $x_{r,1} < \delta$,

BEGIN {*lower bound violation, adjust* $x_{r,1}$ *and recalculate other sublot sizes*}

Set $x_{r,1} = \delta$ and obtain other sublot sizes from the following equations,

$$x_{r,i} = \frac{(1-\delta)\prod_{j=i}^{n-1} z_j}{\sum_{k=1}^{n-1}\prod_{j=k}^{n-1} z_j + 1}, \quad 2 \le i \le n-1,$$

and

$$x_{r,n} = \frac{1-\delta}{\sum_{k=1}^{n-1}\prod_{j=k}^{n-1} z_j + 1}.$$

END-IF {*lower bound violation, adjust* $x_{r,1}$ *and recalculate other sublot sizes*}

Find M_r, length of the longest path, by using $\mathbf{x}_r$.

IF $M_l < M_r$

$u = c_r$

$MinMks = M_l$

$\mathbf{x} = \mathbf{x}_l$

ELSE

$l = c_l$

$MinMks = M_r$

$\mathbf{x} = \mathbf{x}_r$

END-FOR {*perform K iterations to obtain the best z_1 value*}

3. Algorithms obtains the solution vector **x** providing the makespan *MinMks*.

The complexity of the above algorithm is $O(Kn^3)$ since the algorithm employs K iterations each computing n sublot sizes each of which requires $O(n^2)$ computations. The applications of the algorithms presented above yielded very satisfactory results on randomly generated lot streaming problems with attached setups. These results are summarized in the next section.

The performance of the heuristic algorithm proposed for the m x n lot streaming problem with attached setups was tested for several different scenarios. The results of the experiments done for the m x n Attached Setup Algorithm are presented in Table 4.3.

In the experimental analysis, 1050 randomly generated sample problems were used to study the accuracy of the heuristic algorithm designed for the n-sublot lot streaming problems with attached setups. The first part of this study evaluated the effects of the number of iterations on the performance. For this purpose, 600 sample problems were generated having 5 sublots and 10 machines each with processing times uniformly distributed between 5 and 100 time units and setup time to processing time ratios uniformly distributed between 0 and 1. These problems were divided into four groups and each group of problems were solved using the number of iterations of 5, 10, 15 and 20, respectively. The results show that the effect of increasing the iteration number of the

heuristic to increase the performance diminishes after a certain value. In this category, the heuristic provided a satisfactory average performance of 0.87 % error.

Table 4.3 Performance evaluation of the *n*-sublot Bisection - ASA

Problem Size		Range	Range of	Number	Epsilon	# of probs.	Min.	Max.	Average
m	*n*	of p_i	S/P Ratio	of iterations	value	solved	% Error	% Error	% Error
10	5	[5, 100]	[0, 1]	5	0.001	150	0	5.065246	1.181167
10	5	[5, 100]	[0, 1]	10	0.001	150	0	5.130192	0.767988
10	5	[5, 100]	[0, 1]	15	0.001	150	0	5.1224	0.775502
10	5	[5, 100]	[0, 1]	20	0.001	150	0	5.1224	0.772443
					Overall	600	0	5.130192	0.874275
10	5	[5, 100]	[0, 0.30]	10	0.001	50	0	4.201299	0.431312
10	6	[5, 100]	[0, 0.30]	10	0.001	50	7.1E-05	3.25136	0.864265
10	7	[5, 100]	[0, 0.30]	10	0.001	50	0	7.789547	1.148874
					Overall	150	0	7.789547	0.814817
10	5	[5, 100]	[0, 0.50]	10	0.01	50	0	6.495514	1.33297
10	5	[5, 100]	[0, 0.50]	10	0.001	50	0	1.945873	0.428354
10	5	[5, 100]	[0, 0.50]	10	0.0001	50	8.5E-05	1.823285	0.3921
					Overall	150	0	6.495514	0.717808
10	5	[5, 100]	[0, 1]	10	0.001	50	0	2.975425	0.518089
15	5	[5, 100]	[0, 1]	10	0.001	50	0	1.950067	0.57144
20	5	[5, 100]	[0, 1]	10	0.001	50	0	3.251448	0.619426
					Overall	150	0	3.251448	0.569652
				OVERALL		1050	0	7.789547	0.799911

The suitability of applying the *n*-sublot heuristic on the problems with a larger number of sublots was studied in the second part of the experimental analysis. For this part, 50 randomly generated problems were used in each group of 5, 6 and 7 sublots. These problems all had 10 machines, whose setup time to processing time ratios were uniformly distributed between 0 and 0.3. The results of 10 iterations on every problem show that the performance of the heuristic slightly deteriorates as the number of sublots increases. However, the overall average performance was observed as 0.81 % error which is still satisfactory.

The effect of the epsilon values used in the bisection part was studied on 150 sample problems. It was seen that using a smaller epsilon value helps improving the performance, even though it is not as significant as it was on 3-sublot heuristic. An order

of magnitude change in ε has produced approximately 300 % improvement in average error. However, reducing epsilon another order of magnitude did not produce any significant improvement after the first order of magnitude reduction.

The last part of the analysis included the study of the performance of the heuristic on larger machine numbers. 50 sample problems were generated randomly for each group containing 10, 15 and 20 machines, respectively. The results presented in Table 4.3 show that the deterioration of the performance as the number of machine increases is negligible and the heuristic proves to be successful and robust on larger problems.

4.2 Lot Streaming Problems with Transfer Times Explicitly Considered

In the basic lot streaming model, it was assumed that the sublots are transferred instantly between machines. Since some modern manufacturing technologies such as Just-in-Time and Flexible Manufacturing Systems are enhanced with fast transportation equipment serving between work stations, omitting the transportation times spent during manufacturing in the lot streaming model may not cause a considerable amount of error in these environments. However, the time spent to transport work pieces between machines occupies a high percentage of total time spent in the shop in most of the traditional manufacturing environments. Therefore, an improvement in the basic lot streaming model by adding the variable or constant transfer times between machines will be valuable to represent the real conditions in some manufacturing systems.

When it is assumed that there exists a transportation system between two consecutive machines with enough capacity to handle all transfers without delay and each

transfer takes some constant or variable time to be completed instead of being instantaneous, the modified lot streaming formulation will be the following,

(P4.9) minimize T_{mn}

subject to

$$x_1 + x_2 + \ldots + x_n = 1 \tag{4.113}$$

$$T_{11} \geq p_1 x_1 \tag{4.114}$$

$$T_{ij} \geq T_{i,j-1} + p_i x_j, \qquad 1 \leq i \leq m,\ 2 \leq j \leq n \tag{4.115}$$

$$T_{ij} \geq T_{i-1,j} + TT_i + p_i x_j, \qquad 2 \leq i \leq m,\ 1 \leq j \leq n \tag{4.116}$$

$$T_{ij} \geq 0, \qquad 1 \leq i \leq m,\ 1 \leq j \leq n \tag{4.117}$$

$$x_j \geq 0, \qquad 1 \leq j \leq n \tag{4.118}$$

where TT_i is the constant time needed to complete a transfer between machine i-1 and machine i. The above formulation also has a network representation, which is shown in Figure 4.6 below.

Definition 4.5 below states that the paths in the above network can also be represented by sets of crossover indices.

Definition 4.5. *Let $I[j]$, j = 1, 2, …, n-1, denote a machine index in the m x n lot streaming problem with explicit transfer times . The set of machine indices I,*

$$I = \{I[1], I[2], \ldots, I[n-1]\}$$

where $I[j] \leq I[j+1]$, for j = 1, …, n-2, defines the path, denoted by $P(I)$, that crosses over from column j to column j+1 in the network representation of the problem at node $(I[j], j)$, for j = 1, …, n-1. This path, $P(I)$, also traverses the sequence of nodes $(I[j-1], j)$ through $(I[j], j)$, for $j = 1, \ldots, n$, where $I[0] \equiv 1$ and $I[n] \equiv m$.

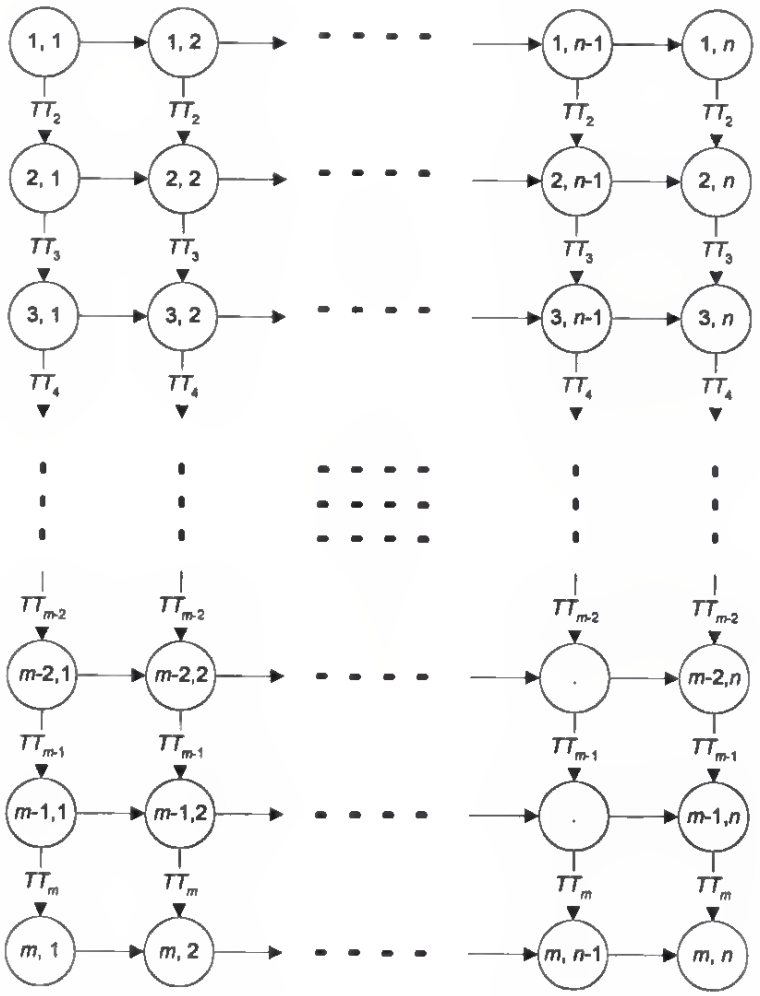

Figure 4.6 Network representation of m x n problem with constant transfer times.

Following Definition 4.5, for a given vector of sublot sizes, the length of the path $P(I)$, denoted by $L_{P(I)}(\mathbf{x})$, is expressed as

$$L_{P(I)}(\mathbf{x}) = \sum_{j=1}^{n} T(I[j-1], I[j])x_j + \sum_{k=2}^{m} TT_k \,, \tag{4.119}$$

where $T(i,j) = \sum_{k=i}^{j} p_k$, p_k is the unit processing time on machine k and TT_k is the constant time needed to move any size of batches from machine k-1 to machine k. Since

every path starts with machine index 1 and ends at machine index m and also follows the same strict order of machines to visit, the constant term $\sum_{k=2}^{m} TT_k$ will be added in the length of every path in the network including the longest path with minimal length which corresponds to the optimal makespan. Therefore, the constant term can be removed from the objective function without altering the structure of the problem. After the sum of the constant transfer times is removed, the remaining part of the length of the path $P(I)$ given in Equation 4.119 will be the same as the case in the basic lot streaming problem with m machines and n sublots presented in Chapter 3. Therefore, the sublot sizes that minimizes the makespan of the basic problem will also minimize the makespan of the problem with transfer times. However, the makespan of the problem with transfer times will be $\sum_{k=2}^{m} TT_k$ time units larger than the makespan of the basic problem.

When the transfer time is not independent of the load size, then instead of the constraint (4.116), the following will be in the formulation,

$$T_{ij} \geq T_{i-1,j} + UT_i x_j + p_i x_j, \quad 2 \leq i \leq m,\ 1 \leq j \leq n \tag{4.120}$$

where UT_i is the time needed to transfer a unit of product between machines i-1 and i. In this case, it is clear that terms $UT_i x_j$ and $p_i x_j$ can be combined in a single term and therefore, the solution techniques for the basic problem presented in Chapter 3 will optimally solve this problem with the modified processing times of p_1, UT_2+p_2, …, UT_n+p_n.

CHAPTER 5
CONCLUSION

In this dissertation, single-job lot streaming problems utilizing consistent and continuous sublots in flow shop environments are discussed. The literature on lot streaming is outlined along with the review of the solution techniques for the $F, 2 \mid S, II \mid n, C, R \mid C_{max}$, $F, 3 \mid S, II \mid n, C, R \mid C_{max}$, and $F, m \mid S, II \mid 2, C, R \mid C_{max}$ problems. The most efficient optimal solution techniques for these problems were introduced by Baker [Bak87], Glass et al. [Gla94] and Williams and Tüfekçi [Wil95], respectively.

The first contribution in this dissertation is the study of the $F, m \mid S, II \mid 3, C, R \mid C_{max}$ problem. This problem is solved through the project network representation of the problem. The solution procedure developed locates the longest path with minimal length which corresponds to the optimal makespan in the network representation in $O(m^2)$ time. It is also shown by numerical examples that the solution of the 3-sublot problem can frequently offer a good approximation to the solution of the n-sublot problem. Based on observations some heuristic approaches are proposed.

The efforts to optimally solve the $F, m \mid S, II \mid n, C, R \mid C_{max}$ problem consist a large part of this dissertation. Similar to the case in the 3-sublot problem, the network representation offered several properties whose utilization leads to the development of efficient and optimal solution methods. This dissertation includes two optimal solution

algorithms for the n-sublot problem whose complexity is $O(n^3*(m+n))$ and $O(n^2*(m+n))$, respectively.

Additionally, the basic lot streaming problem is studied in terms of its modifiability to reflect some common features of real manufacturing environments. First step in this process is the addition of the setup activities in the lot streaming model. Setup activities are classified into two groups, i.e., attached and detached setups, depending on whether or not they require some work pieces to arrive from upstream machines at the work station that will be prepared for processing. In this dissertation, only the pure cases where all the machines have the same type of setups are considered. It is seen that two sublot problems in both types of setups, i.e., $F, m \mid S, II, DS \mid 2, C, R \mid C_{max}$ and $F, m \mid S, II, AS \mid 2, C, R \mid C_{max}$, are optimally solvable through the existing solution techniques for the same size basic lot streaming problem. When the sublot number is three or larger, the problems are solved by some approximation methods since the network representation of the lot streaming problem with setups does not provide any monotonic property that will guarantee locating the optimal solution. However, an extensive computational analysis shows that each heuristic provides near-optimal and often optimal results for the type of problem it was designed to solve. It is also shown that the solution procedures developed in this dissertation for the basic lot streaming problem are also applicable to the problems considering constant or variable transfer times between work stations.

Therefore, the solution procedures developed during the course of this study provide very robust results for the single job lot streaming problems in flow shop environments. Regarding the enhanced lot streaming model, the solution techniques have a practical value as well.

REFERENCES

[Bak87] Baker, K.R., "Lot streaming to reduce cycle time in a flow shop," Working Paper No. 203, The Amos Tuck School of Business Administration, Dartmouth College, Hanover, NH 03755, June 1987.

[Bak90] Baker, K.R., and D.F. Pyke, "Solution procedures for the lot streaming problem," Decision Sciences, 21 (1990), 475-491.

[Bak93] Baker, K.R., and D. Jia, "A comparative study of lot streaming procedures," Omega International Journal of Management Science, 21 (1993), 561-566.

[Bak95] Baker, K.R., "Lot streaming in the two-machine flow shop with setup times," Annals of Operations Research, 57 (1995), 1-11.

[Ben94] Benli, Ö.S., "Lot streaming in production scheduling: Streaming a single job in a flow shop," Research Report IEOR-9409, Department of Industrial Engineering, Bilkent University, Ankara, Turkey, December 1994.

[Che94] Chen, J., and G. Steiner, "Lot streaming with attached setups in three-machine flow shops," Working Paper, School of Business, McMaster University, Hamilton, Ontario, Canada, 1994.

[Che95] Chen, J., and G. Steiner, "Lot streaming with detached setups in three-machine flow shops," Working Paper # 399, School of Business, McMaster University, Hamilton, Ontario, Canada, 1995.

[Cet92] Çetinkaya, F.C., and M.S. Kayalıgil, "Unit sized transfer batch scheduling with setup times," Computers and Industrial Engineering, 22 (1992), 177-183.

[Cet94a] Çetinkaya, F.C., and J.N.D. Gupta, "Flow shop lot streaming to minimize total weighted flow time," Research Memorandum No. 94-24, School of Industrial Engineering, Purdue University, West Lafayette, IN 47907, June 1994.

[Cet94b] Çetinkaya, F.C., "Lot streaming in a two-stage flow shop with set-up, processing and removal times separated," Journal of Operational Research Society, 45 (1994), 1445-1455.

[Cox97] Cox, J.F., and J.H. Blackstone, "Managing by the Theory of Constraints," Production and Inventory Control Handbook, J.H. Greene (ed.), McGraw-Hill, 3rd ed., 1997.

[Dau93] Dauzère-Péres, S., and J.-B. Lasserre, "An iterative procedure for lot streaming in job-shop scheduling," Computers and Industrial Engineering, 25 (1993), 231-234.

[Gla94] Glass, C.A., J.N.D. Gupta, and C.N. Potts, "Lot streaming in three-stage production processes," European Journal of Operational Research, 75 (1994), 378-394.

[Gol92] Goldratt, E.M., and J. Cox, "The Goal: A Process of Ongoing Improvement," North River Press, 2nd Revised Edition, 1992.

[Goy76] Goyal, S.K., "Note on 'Manufacturing cycle time determination for a multi-stage economic production quantity model,'" Management Science, 23 (1976), 332-333.

[Gra86] Graves, S.C., and M.M. Kostreva, "Overlapping Operations in Material Requirements Planning," Journal of Operations Management, 6 (1986), 283-294.

[Han71] Hancock, T.M., "Effects of lot-splitting under various routing strategies," International Journal of Production Research, 9 (1971), 68-74.

[Jac88] Jacobs, F.R., and D.J. Bragg, "Repetitive lots: Flow-time reductions through sequencing and dynamic batch sizing," Decision Sciences, 19 (1988), 281-294.

[Joh54] Johnson, S.M., "Optimal two- and three-stage production schedules with set-up times included," Naval Research Logistics Quarterly, 1 (1954), 61-68.

[Kro90] Kropp, D.H., and T.L. Smunt, "Optimal and heuristic models for lot splitting in a flow shop," Decision Sciences, 21 (1990), 691-709.

[Meg83] Megiddo, N., "Linear-Time Algorithms for Linear Programming in $\Re^3$ and Related Problems," Society for Industrial and Applied Mathematics, 12 (1983), 759-776.

[Pin95] Pinedo, M., "Scheduling: Theory, Algorithms and Systems," Prentice Hall, Englewood Cliffs, New Jersey, 1995.

[Pot89] Potts, C.N., and K.R. Baker, "Flow shop scheduling with lot streaming," Operations Research Letters, 8 (1989), 297-303.

[Pot92] Potts, C.N., and L.N. Van Wassenhove, "Integrating scheduling with batching and lot sizing: a review of algorithms and complexity," Journal of Operational Research Society, 1992.

[Rei66] Reiter, S., "A system for managing job shop production," Journal of Business, 34 (1966), 371-393.

[Sen94] Şen, A., E. Topaloğlu, Ö.S. Benli, "Optimal streaming of a single job in a two stage flow shop," Research Report IEOR-9413, Department of Industrial Engineering, Bilkent University, Ankara, Turkey, November 1994.

[Ste93] Steiner, G., and W.G. Truscott, "Batch scheduling to minimize cycle time, flow time, and processing cost," IIE Transactions, 25 (1993), 90-96.

[Sze75] Szendrovits, A.Z., "Manufacturing cycle time determination for a multi-stage economic production quantity model," Management Science, 22 (1975), 298-308.

[Sze76] Szendrovits, A.Z., "On the optimality of sub-batch sizes for a multi-stage EPQ model - A rejoinder," Management Science, 23 (1976), 334-337.

[Top94] Topaloğlu, E., A. Şen, and Ö.S. Benli, "Optimal streaming of a single job in an m stage flow shop with two sublots," Research Report IEOR-9412, Department of Industrial Engineering, Bilkent University, Ankara, Turkey, November 1994.

[Tri87] Trietsch, D., "Optimal transfer lots for batch manufacturing: A basic case and extensions," Working Paper, Department of Administrative Sciences, Naval Postgraduate School, Monterey, CA 93943, November 1987.

[Tri89] Trietsch, D., "Polynomial transfer lot sizing techniques for batch processing on consecutive machines," Working Paper, Department of Administrative Sciences, Naval Postgraduate School, Monterey, CA 93943, September 1989.

[Tri92] Trietsch, D., and K.R. Baker "Basic techniques for lot streaming," Working Paper No. 262, The Amos Tuck School of Business Administration, Dartmouth College, Hanover, NH 03755, August 1992.

[Tru85] Truscott, W.G., "Scheduling production activities in multi-stage batch processing manufacturing systems," International Journal of Production Research, 23 (1985), 315-328.

[Tru86] Truscott, W.G., "Production scheduling with capacity-constrained transportation activities," Journal of Operations Management, 6 (1986), 333-348.

[Vic92] Vickson, R.G., and B.E. Alfredsson, "Two- and three-machine flow shop scheduling problems with equal sized transfer batches," International Journal of Production Research, 30 (1992), 1551-1574.

[Vic95] Vickson, R.G., "Optimal lot streaming for multiple products in a two-machine flow shop," European Journal of Operational Research, 85 (1995), 556-575.

[Wil92] Williams, E.F., and S. Tüfekçi, "Polynomial time algorithms for the *m*x2 lot streaming problem," Research Report No. 92-10, Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL 32611, July 1992.

[Wil95] Williams, E.F., and S. Tüfekçi, "A linear time algorithm for the 2 sublot lot streaming problem," Research Report No. 95-2, Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL 32611, January 1995.

[Wil97] Williams, E.F., S. Tüfekçi, and M. Akansel "O(m^2) algorithms for the 2 and 3 sublot lot streaming problem," Production and Operations Management, 6 (1997), 74-96.

BIOGRAPHICAL SKETCH

Mehmet Akansel was born in Düzce, Bolu, Turkey, on June 12, 1971. He received his B.S. in Industrial Engineering from İstanbul Technical University, İstanbul, Turkey, in June 1991. He also received his M.S. in Industrial Engineering from the same institution in February 1993. He worked as a teaching and research assistant in the Department of Industrial Engineering at Uludağ University, Bursa, Turkey, from May 1992 to December 1993. In January 1994, he started his Ph.D. studies in the Department of Industrial and Systems Engineering at the University of Florida. He was sponsored by the Turkish government throughout his Ph.D. studies with the condition that he work for Uludağ University upon completion. He is a member of INFORMS and IIE. He is married and has a daughter.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Süleyman Tüfekçi, Chairman
Associate Professor of Industrial
and Systems Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Sherman X. Bai
Assistant Professor of Industrial
and Systems Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Ş. Selçuk Erengüç
Professor of Decision and Information
Sciences

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Asoo J. Vakharia
Associate Professor of Decision and
Information Sciences

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Sencer Yeralan
Associate Professor of Industrial
and Systems Engineering

This dissertation was submitted to the Graduate Faculty of the College of Engineering and to the Graduate School and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

May, 1998

Winfred M. Phillips
Dean, College of Engineering

Karen A. Holbrook
Dean, Graduate School

LD
1780
1994
.A313

UNIVERSITY OF FLORIDA
3 1262 06554 9086